

International Series in  
Operations Research & Management Science

Miguel F. Anjos  
Jean B. Lasserre *Editors*

# Handbook on Semidefinite, Conic and Polynomial Optimization



 Springer

# **International Series in Operations Research & Management Science**

**Volume 166**

Series Editor

Frederick S. Hillier  
Stanford University, CA, USA

Special Editorial Consultant

Camille C. Price  
Stephen F. Austin State University, TX, USA

For further volumes:  
<http://www.springer.com/series/6161>



Miguel F. Anjos • Jean B. Lasserre  
Editors

# Handbook on Semidefinite, Conic and Polynomial Optimization



*Editors*

Miguel F. Anjos

Department of Mathematics and Industrial  
Engineering & GERAD

École Polytechnique de Montréal  
Montréal, QC, Canada H3C 3A7  
[anjos@stanfordalumni.org](mailto:anjos@stanfordalumni.org)

Jean B. Lasserre

LAAS-CNRS and Institute of Mathematics  
7 Avenue du Colonel Roche  
31077 Toulouse Cedex 4  
France  
[lasserre@laas.fr](mailto:lasserre@laas.fr)

ISSN 0884-8289

ISBN 978-1-4614-0768-3

e-ISBN 978-1-4614-0769-0

DOI 10.1007/978-1-4614-0769-0

Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2011938887

© Springer Science+Business Media, LLC 2012

All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

Printed on acid-free paper

Springer is part of Springer Science+Business Media ([www.springer.com](http://www.springer.com))

# Preface

Conic optimization is a significant and thriving research area within the optimization community. Conic optimization is the general class of problems concerned with optimizing a linear function over the intersection of an affine space and a closed convex cone. One special case of great interest is the choice of the cone of positive semidefinite matrices for which the resulting optimization problem is called a semidefinite optimization problem.

Semidefinite optimization, or semidefinite programming (SDP), has been studied (under different names) since at least the 1940s. Its importance grew immensely during the 1990s after polynomial-time interior-point methods for linear optimization were extended to solve SDP problems (and more generally, to solve convex optimization problems with efficiently computable self-concordant barrier functions). Some of the earliest applications of SDP that followed this development were the solution of linear matrix inequalities in control theory, and the design of polynomial-time approximation schemes for hard combinatorial problems such as the maximum-cut problem.

This burst of activity in the 1990s led to the publication of the *Handbook of Semidefinite Programming* in the year 2000. That Handbook, edited by Wolkowicz, Saigal, and Vandenberghe, provided an overview of much of the activity in the area.

Research into semidefinite programming has continued unabated, and a new development since the beginning of the twenty-first century has been the fruitful interaction with algebraic geometry through the close connections between semidefinite matrices and polynomial optimization problems. This has brought about several important new results and led to an even higher level of research activity. Much of this activity can be followed on the *Optimization Online* (<http://www.optimization-online.org>) and *ArXiv* (<http://arxiv.org>) websites.

The objective of this *Handbook on Semidefinite, Conic and Polynomial Optimization* is to provide the reader with a snapshot of the state of the art in the growing and mutually enriching areas of semidefinite optimization, conic optimization, and polynomial optimization. Our intention is to provide a compendium of the research activity that has taken place since the publication of the seminal Handbook

mentioned above. It is our hope that this will motivate more researchers, especially doctoral students and young graduates, to become involved in these thrilling areas of optimization.

## Overview of the Handbook

The Handbook begins with a chapter presenting the basics of semidefinite, conic, and polynomial optimization. The subsequent 30 chapters are grouped into four parts: *Theory*, *Algorithms*, *Software*, and *Applications*.

### ***Theory***

This first part represents approximately one-third of the Handbook. It covers many significant theoretical developments, and several chapters reflect the interactions between conic optimization and polynomial optimization.

### ***Algorithms***

This second part documents a number of different directions in which the development of algorithms is taking place. It indicates the breadth of approaches being applied to solve conic optimization problems, including both interior-point methods and more recent approaches.

### ***Software***

It is a sign of the maturity of the field that there are now many software packages to solve small- and medium-sized semidefinite optimization problems. The first chapter of this part provides an overview of the state of the art, while the subsequent chapters document the latest developments in three commonly used software packages.

There are also a number of interfaces that facilitate the use of conic optimization software. We have chosen not to include these in the Handbook in order to keep the focus on the theoretical and algorithmic concepts behind the solvers, and thus to help guide the reader to the most appropriate approaches for specific applications.

Like all other aspects of the field, the software offerings are in constant evolution. As a starting point for the interested reader, we provide the URL for the software section of the *Semidefinite Programming* webpage maintained by Christoph Helmberg: [http://www-user.tu-chemnitz.de/~helmberg/sdp\\_software.html](http://www-user.tu-chemnitz.de/~helmberg/sdp_software.html).

## ***Applications***

Finally, the fourth part is concerned with some of the application areas where conic optimization has made a significant impact in recent years. Several of these involve hard combinatorial optimization problems that continue to benefit from the advances in theory, algorithms, and software mentioned in the previous parts.

**Acknowledgements** This Handbook benefited tremendously from the generous help of all those who kindly agreed to referee one or more chapters, assisting us and the authors to significantly improve the content. They were: Kurt Anstreicher, Michel Baes, Frédéric Bonnans, Valentin Brimkov, Samuel Burer, Brian Borchers, Chek-Beng Chua, Mirjam Dür, Alexander Engau, Anja Fischer, Mituhiro Fukuda, João Gouveia, Luigi Grippo, Stefano Gualandi, Christoph Helmberg, Didier Henrion, Philipp Hungerländer, Michal Kocvara, Nathan Krislock, William Martin, John Mitchell, Baback Moghaddam, Jiawang Nie, Javier Peña, Veronica Piccialli, Daniel Plaumann, Mihai Putinar, Houduo Qi, Grant Schoenebeck, Frank Sottile, David Steurer, Hajime Tanaka, Thorsten Theobald, and Manuel Vieira.

We are also grateful for the support of GERAD in the compilation of this Handbook, and particularly the hard work and dedication of Ms. Marilyne Lavoie.

Finally, financial support from the Alexander von Humboldt Foundation and the Natural Sciences and Engineering Research Council of Canada is gratefully acknowledged by the first editor.

Montréal and Toulouse

Miguel F. Anjos  
Jean B. Lasserre



# Contents

<b>1</b>	<b>Introduction to Semidefinite, Conic and Polynomial Optimization ...</b>	<b>1</b>
	Miguel F. Anjos and Jean B. Lasserre	
<b>Part I Theory</b>		
<b>2</b>	<b>The Approach of Moments for Polynomial Equations .....</b>	<b>25</b>
	Monique Laurent and Philipp Rostalski	
<b>3</b>	<b>Algebraic Degree in Semidefinite and Polynomial Optimization.....</b>	<b>61</b>
	Kristian Ranestad	
<b>4</b>	<b>Semidefinite Representation of Convex Sets and Convex Hulls .....</b>	<b>77</b>
	J. William Helton and Jiawang Nie	
<b>5</b>	<b>Convex Hulls of Algebraic Sets .....</b>	<b>113</b>
	João Gouveia and Rekha Thomas	
<b>6</b>	<b>Convex Relaxations and Integrality Gaps .....</b>	<b>139</b>
	Eden Chlamtac and Madhur Tulsiani	
<b>7</b>	<b>Relaxations of Combinatorial Problems Via Association Schemes ...</b>	<b>171</b>
	Etienne de Klerk, Fernando M. de Oliveira Filho, and Dmitrii V. Pasechnik	
<b>8</b>	<b>Copositive Programming .....</b>	<b>201</b>
	Samuel Burer	
<b>9</b>	<b>Invariant Semidefinite Programs .....</b>	<b>219</b>
	Christine Bachoc, Dion C. Gijswijt, Alexander Schrijver, and Frank Vallentin	
<b>10</b>	<b>A “Joint+Marginal” Approach in Optimization .....</b>	<b>271</b>
	Jean B. Lasserre	

<b>11</b>	<b>An Introduction to Formally Real Jordan Algebras and Their Applications in Optimization .....</b>	297
	F. Alizadeh	
<b>12</b>	<b>Complementarity Problems Over Symmetric Cones: A Survey of Recent Developments in Several Aspects .....</b>	339
	Akiko Yoshise	
<b>13</b>	<b>Convexity and Semidefinite Programming in Dimension-Free Matrix Unknowns .....</b>	377
	J. William Helton, Igor Klep, and Scott McCullough	
<b>14</b>	<b>Positivity and Optimization: Beyond Polynomials .....</b>	407
	Jean B. Lasserre and Mihai Putinar	

## Part II Algorithms

<b>15</b>	<b>Self-Regular Interior-Point Methods for Semidefinite Optimization .....</b>	437
	Maziar Salahi and Tamás Terlaky	
<b>16</b>	<b>Elementary Optimality Conditions for Nonlinear SDPs .....</b>	455
	Florian Jarre	
<b>17</b>	<b>Recent Progress in Interior-Point Methods: Cutting-Plane Algorithms and Warm Starts .....</b>	471
	Alexander Engau	
<b>18</b>	<b>Exploiting Sparsity in SDP Relaxation of Polynomial Optimization Problems .....</b>	499
	Sunyoung Kim and Masakazu Kojima	
<b>19</b>	<b>Block Coordinate Descent Methods for Semidefinite Programming .....</b>	533
	Zaiwen Wen, Donald Goldfarb, and Katya Scheinberg	
<b>20</b>	<b>Projection Methods in Conic Optimization .....</b>	565
	Didier Henrion and Jérôme Malick	
<b>21</b>	<b>SDP Relaxations for Non-Commutative Polynomial Optimization ...</b>	601
	Miguel Navascués, Stefano Pironio, and Antonio Acín	
<b>22</b>	<b>Semidefinite Programming and Constraint Programming .....</b>	635
	Willem-Jan van Hoeve	

## Part III Software

<b>23</b>	<b>The State-of-the-Art in Conic Optimization Software .....</b>	671
	Hans D. Mittelmann	

<b>24 Latest Developments in the SDPA Family for Solving Large-Scale SDPs .....</b>	687
Makoto Yamashita, Katsuki Fujisawa, Mituhiro Fukuda, Kazuhiro Kobayashi, Kazuhide Nakata, and Maho Nakata	
<b>25 On the Implementation and Usage of SDPT3 – A MATLAB Software Package for Semidefinite-Quadratic-Linear Programming, Version 4.0 .....</b>	715
Kim-Chuan Toh, Michael J. Todd, and Reha H. Tütüncü	
<b>26 PENNON: Software for Linear and Nonlinear Matrix Inequalities ..</b>	755
Michal Kocvara and Michael Stingl	

#### Part IV Applications

<b>27 SDP Relaxations for Some Combinatorial Optimization Problems...</b>	795
Renata Sotirov	
<b>28 Computational Approaches to Max-Cut .....</b>	821
Laura Palagi, Veronica Piccialli, Franz Rendl, Giovanni Rinaldi, and Angelika Wiegele	
<b>29 Global Approaches for Facility Layout and VLSI Floorplanning .....</b>	849
Miguel F. Anjos and Frauke Liers	
<b>30 Euclidean Distance Matrices and Applications .....</b>	879
Nathan Krislock and Henry Wolkowicz	
<b>31 Sparse PCA: Convex Relaxations, Algorithms and Applications .....</b>	915
Youwei Zhang, Alexandre d'Aspremont, and Laurent El Ghaoui	
<b>Index .....</b>	941



# Chapter 1

## Introduction to Semidefinite, Conic and Polynomial Optimization

Miguel F. Anjos and Jean B. Lasserre

### Part I: Conic Optimization

Conic optimization refers to the problem of optimizing a linear function over the intersection of an affine space and a closed convex cone. We focus particularly on the special case where the cone is chosen as the cone of positive semidefinite matrices for which the resulting optimization problem is called a semidefinite optimization problem.

#### 1.1 Semidefinite Programming

Semidefinite optimization, or semidefinite programming (SDP), refers to the class of optimization problems where a linear function of a symmetric matrix variable  $X$  is optimized subject to linear constraints on the elements of  $X$  and the additional constraint that  $X$  must be positive semidefinite. This includes linear programming (LP) problems as a special case, namely when all the matrices involved are diagonal.

SDP problems are an important class of optimization problems for several reasons. First, SDP problems are solvable in polynomial time. This means that any problem that can be expressed using SDP is also solvable in polynomial time.

---

M.F. Anjos (✉)

Department of Mathematics and Industrial Engineering and GERAD,  
École Polytechnique de Montréal, Montréal, QC, Canada H3C 3A7  
e-mail: [anjos@stanfordalumni.org](mailto:anjos@stanfordalumni.org)

J.B. Lasserre

LAAS-CNRS and Institute of Mathematics, 7 Avenue du Colonel Roche,  
31077 Toulouse Cédex 4, France  
e-mail: [lasserre@laas.fr](mailto:lasserre@laas.fr)

Second, SDP problems can be solved efficiently in practice. This can be done by using one of the software packages available, or alternatively by implementing a suitable algorithm. Third, SDP can be used to obtain tight approximations for hard problems in integer and global optimization. Fourth, SDP problems are applicable to a wide range of practical applications in areas such as control theory, circuit design, sensor network localization, and principal component analysis.

SDP has a number of similarities with LP. Like LP problems, SDP problems also come in pairs. One of the problems is referred to as the *primal* problem, and the second one is the *dual* problem. Either problem can be chosen as the primal, since the two problems are dual to each other. The most common standard formulation of SDP is as follows:

$$\begin{array}{ll} \text{(P) } \inf \langle C, X \rangle & \text{(D) } \sup b^T y \\ \text{s.t. } \langle A_i, X \rangle = b_i, i = 1, \dots, m & \text{s.t. } \sum_{i=1}^m y_i A_i + S = C \\ X \succeq 0 & S \succeq 0 \end{array} \quad (1.1)$$

where (P) denotes the primal problem, and (D) the dual problem; the variables  $X$  and  $S$  are in  $\mathcal{S}^n$ , the space of  $n \times n$  real symmetric matrices;  $X \succeq 0$  denotes that the matrix  $X$  is positive semidefinite; the data matrices  $A_i$  and  $C$  may be assumed to be symmetric without loss of generality; and  $b \in \mathbb{R}^m$  and  $y \in \mathbb{R}^m$  are column vectors. We use the inner product between two matrices in  $\mathcal{S}^n$  defined as

$$\langle R, S \rangle := \text{trace}(RS) = \sum_{i=1}^n \sum_{j=1}^n R_{i,j} S_{i,j}$$

where  $\text{trace } M$  denotes the trace of the square matrix  $M$ , which is the sum of the diagonal elements of  $M$ . It is normally assumed that, without loss of generality, the matrices  $A_i$ ,  $i = 1, \dots, m$ , are linearly independent. When it is known from the context that the optimal values are attained, it is common to replace  $\inf$  by  $\min$  and  $\sup$  by  $\max$ .

The dual SDP problem can equivalently be written without using the dual variable  $S$ :

$$\begin{array}{ll} \sup b^T y & \\ \text{s.t. } C - \sum_{i=1}^m y_i A_i \succeq 0. & \end{array}$$

The resulting constraint is called a *linear matrix inequality (LMI)*. The inequality is interpreted according to the partial order induced on  $\mathcal{S}^n$  by the positive semidefinite cone. This is called the Löwner partial order, and is defined as follows:

$$A \succeq B \iff A - B \succeq 0.$$

Before proceeding, we illustrate via two examples how SDP can be applied to different types of problems. The concepts illustrated in these examples are simple yet extremely useful. The first example is concerned with eigenvalue optimization.

*Example 1.1 (Eigenvalue optimization).* Eigenvalues of a matrix variable are important in many contexts, both theoretical and practical. Suppose we wish to minimize the maximum eigenvalue of a symmetric matrix  $A(z)$  whose entries depend linearly on the variable  $z$ :

$$A(z) = A_0 + z_1 A_1 + z_2 A_2 + \dots + z_\ell A_\ell$$

for some  $\ell \geq 1$ . The problem of interest is thus

$$\min_{z \in \Re} \lambda_{\max}(A(z)). \quad (1.2)$$

To formulate this problem using SDP, we make use of the fact that a symmetric matrix is positive semidefinite if and only if all its (real) eigenvalues are non-negative (see Theorem 1.3 below). First we observe that (1.2) is equivalent to

$$\begin{aligned} & \min t \\ \text{s.t. } & t \geq \lambda_{\max}(A(z)). \end{aligned}$$

Since  $t \geq \lambda_{\max}(A(z))$  is equivalent to  $\lambda_{\min}(tI - A(z)) \geq 0$ , by Theorem 1.3, this constraint is equivalent to  $tI - A(z) \geq 0$ . Hence, problem (1.2) is equivalent to

$$\begin{aligned} & \max -t \\ \text{s.t. } & tI - A_0 - z_1 A_1 - z_2 A_2 - \dots - z_\ell A_\ell \geq 0, \end{aligned}$$

which is an SDP problem in the dual form above.

By a similar reasoning, the problem of minimizing the spectral norm of  $A(z)$ :

$$\min_{z \in \Re} |\lambda_{\max}(A(z))|$$

can be written as the following SDP problem:

$$\begin{aligned} & \max -t \\ \text{s.t. } & tI - A_0 - z_1 A_1 - z_2 A_2 - \dots - z_\ell A_\ell \geq 0 \\ & tI + A_0 + z_1 A_1 + z_2 A_2 + \dots + z_\ell A_\ell \geq 0. \end{aligned}$$

The second example shows two derivations of an SDP relaxation for the maximum-cut (max-cut) problem. Following the ground-breaking work of Goemans and Williamson [4], max-cut has become one of the flagship applications of SDP.

*Example 1.2 (A basic SDP relaxation for max-cut).* The max-cut problem is a combinatorial optimization problem on undirected graphs with weights on the edges. Given such a graph  $G = (V, E)$  with node set  $V$  and edge set  $E$ , the problem consists in finding a partition of  $V$  into two parts so as to maximize the sum of the weights on the edges that are cut by the partition. (An edge is said to be cut if it has

exactly one end on each side of the partition.) Without loss of generality, we assume that  $G$  has no loops, and that it is a complete graph since non-edges can be added in with zero weight to complete the graph without changing the problem.

Let the given graph  $G$  have node set  $V = \{1, \dots, n\}$  and let it be described by its weighted adjacency matrix  $W = (w_{ij})$ . Let the vector  $v \in \{-1, 1\}^n$  represent any cut in the graph via the interpretation that the sets  $\{i : v_i = +1\}$  and  $\{i : v_i = -1\}$  specify a partition of the node set of the graph. Then max-cut can be formulated as:

$$\begin{aligned} & \max \sum_{j=2}^n \sum_{i=1}^j w_{ij} \left( \frac{1 - v_i v_j}{2} \right) \\ & \text{s.t. } v_i^2 = 1, \quad i = 1, \dots, n, \end{aligned} \tag{1.3}$$

so that the term multiplying  $w_{ij}$  in the sum equals one if the edge  $(i, j)$  is cut, and zero otherwise. Since  $w_{ij} = w_{ji}$ ,

$$\sum_{j=2}^n \sum_{i=1}^j w_{ij} \left( \frac{1 - v_i v_j}{2} \right) = \sum_{j=1}^n \sum_{i=1}^n w_{ij} \left( \frac{1 - v_i v_j}{4} \right),$$

and so the objective function can be expressed as:

$$\begin{aligned} \frac{1}{4} \sum_{j=1}^n \sum_{i=1}^n (1 - v_i v_j) w_{ij} &= \frac{1}{4} \sum_{i=1}^n \left( \sum_{j=1}^n w_{ij} v_i^2 - \sum_{j=1}^n w_{ij} v_i v_j \right) \\ &= \frac{1}{4} \left( \sum_{i=1}^n \left( \sum_{j=1}^n w_{ij} \right) v_i v_i - \sum_{i=1}^n \sum_{j=1}^n v_i v_j \right) \\ &= \frac{1}{4} (v^T \text{Diag}(We)v - v^T W v) \\ &= \frac{1}{4} v^T L v, \end{aligned}$$

where  $L := \text{Diag}(We) - W$  denotes the Laplacian matrix associated with the graph,  $e$  denotes the vector of all ones, and  $\text{Diag}$  denotes a diagonal matrix with its diagonal formed from the vector given as its argument. We can thus rewrite (1.3) as:

$$\begin{aligned} & \max \quad v^T Q v \\ & \text{s.t.} \quad v_i^2 = 1, \quad i = 1, \dots, n, \end{aligned}$$

where  $Q := \frac{1}{4} L$ .

To obtain an SDP relaxation, we now formulate max-cut in  $\mathcal{S}^n$ . Consider the change of variable  $X = vv^T, v \in \{-1, 1\}^n$ . Then  $X \in \mathcal{S}^n$  and  $v^T Q v = \text{trace } v^T Q v =$

trace  $QX$  (using the fact that trace  $AB = \text{trace } BA$ ), and it can be shown (see e.g. [28]) that max-cut is equivalent to

$$\begin{aligned} & \max \langle Q, X \rangle \\ \text{s.t. } & X_{i,i} = 1, i = 1, \dots, n \\ & \text{rank}(X) = 1 \\ & X \succeq 0, X \in \mathcal{S}^n. \end{aligned} \tag{1.4}$$

Removing the rank constraint (which is not convex) gives the basic SDP relaxation:

$$\begin{aligned} & \max \langle Q, X \rangle \\ \text{s.t. } & \text{diag}(X) = e \\ & X \succeq 0, \end{aligned} \tag{1.5}$$

where  $\text{diag}$  denotes a vector containing the diagonal entries of the matrix argument. The dual SDP problem is

$$\begin{aligned} & \min e^T y \\ \text{s.t. } & S = \text{Diag}(y) - Q \\ & S \succeq 0. \end{aligned} \tag{1.6}$$

An alternative way to relax max-cut to an SDP is to replace the requirement that  $v_i \in \{-1, 1\}$  by  $v_i \in \mathbb{R}^p$ , where  $p \geq 1$ . The resulting formulation of max-cut is:

$$\begin{aligned} & \max \text{trace } V Q V^T \\ \text{s.t. } & \|v_i\|_2 = 1, i = 1, \dots, n \\ & |v_i^T v_j| = 1, i = 1, \dots, n, j = i + 1, \dots, n, \end{aligned} \tag{1.7}$$

where  $V = (v_1, \dots, v_n)$ . Note that  $p$  does not appear explicitly in the formulation.

The matrix variable  $X$  for the SDP relaxation is now obtained by letting  $X_{i,j} := v_i^T v_j$ ,  $i, j = 1, \dots, n$ . Since

$$X = V^T V, \|v_i\|_2 = 1 \forall i \Leftrightarrow \text{diag}(X) = e, X \succeq 0$$

(see Theorem 1.5 below), problem (1.7) can be rewritten as:

$$\begin{aligned} & \max \langle Q, X \rangle \\ \text{s.t. } & \text{diag}(X) = e \\ & X \succeq 0 \\ & |X_{ij}| = 1, i = 1, \dots, n, j = i + 1, \dots, n. \end{aligned} \tag{1.8}$$

Removing the last set of constraints (which is also non-convex) yields the SDP relaxation (1.5) once more.

It is clear that the SDP problem is a relaxation of the original problem. Thus, the optimal value of the SDP problem provides a global upper bound on the true optimal value of the combinatorial problem. Goemans and Williamson showed that for graphs with non-negative edge weights, the SDP relaxation has an optimal value that is at most 14% greater than the value of the global optimal cut [4]. The proof of this result makes use of the matrix  $V$  in (1.7).

In this last example, we illustrated two different, but equivalent, ways of deriving (and interpreting) SDP relaxations for combinatorial problems:

1. Embed the vector of binary variables into a rank-one matrix, formulate the combinatorial problem exactly using the entries of this matrix, and then remove the rank constraint to obtain an SDP; or
2. Replace the binary variables with real vectors of a suitably chosen length, and interpret their pairwise inner products as entries in a positive semidefinite matrix.

For more details on different ways to obtain SDP relaxations of max-cut and other combinatorial optimization problems, see [28].

## 1.2 Positive Semidefinite Matrices

We begin with the definition of positive semidefiniteness.

**Definition 1.1.** A matrix  $A \in \mathcal{S}^n$  is said to be *positive semidefinite (psd)* if

$$y^T A y = \sum_{j=1}^n \sum_{i=1}^n A_{i,j} y_i y_j \geq 0 \text{ for all } y \in \mathbb{R}^n.$$

When the condition holds with strict positivity for all  $y \neq 0$ ,  $A$  is said to be *positive definite (pd)*.

We use  $A \succeq 0$  to denote that  $A$  is psd, and  $A > 0$  for  $A$  pd.

We also use  $\mathcal{S}_+^n$  (resp.  $\mathcal{S}_{++}^n$ ) to denote the set of psd (resp. pd) matrices.

*Example 1.3.* The matrix  $A = \begin{pmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{pmatrix}$  is a feasible solution for (1.4) and (1.5).

To verify this, observe that  $A = vv^T$  with  $v = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$ . Hence,  $y^T A y = (y^T v)^2 \geq 0$  for all  $y \in \mathbb{R}^n$ .

The  $3 \times 3$  identity matrix is feasible for (1.5), but not for (1.4), and it is easy to check that it is pd.

It follows immediately from Definition 1.1 that:

- Every non-negative linear combination of psd matrices is psd:

$$y^T \left( \sum_j \lambda_j X_j \right) y = \sum_j \lambda_j (y^T X_j y) \geq 0$$

whenever  $\lambda_j \geq 0, X_j \geq 0$  for all  $j$ .

- If  $\lambda_j \geq 0, \sum_j \lambda_j > 0, X_j > 0$  for all  $j$ , then the linear combination  $\sum_j \lambda_j X_j$  satisfies

$$y^T \left( \sum_j \lambda_j X_j \right) y = \sum_j \lambda_j (y^T X_j y) > 0$$

for all  $y \neq 0$ , and hence is pd.

- If  $X, Y \in \mathcal{S}_+^n$ , then  $\lambda X + (1 - \lambda)Y \in \mathcal{S}_+^n$  for all  $0 \leq \lambda \leq 1$ . The same holds for  $\mathcal{S}_{++}^n$ , and thus both are convex sets. Note however that only  $\mathcal{S}_+^n$  is closed.

Consider the following definition:

**Definition 1.2.** A subset  $\mathcal{K} \subset \mathcal{S}^n$  is called a *cone* if it is closed under positive scalar multiplication, i.e.

$$\lambda X \in \mathcal{K} \text{ whenever } X \in \mathcal{K} \text{ and } \lambda > 0.$$

Note that the origin may or may not be included. Clearly, both  $\mathcal{S}_+^n$  and  $\mathcal{S}_{++}^n$  are convex cones. The cone  $\mathcal{S}_+^n$  further possesses the following properties:

**Definition 1.3.** A cone  $\mathcal{K}$  is *pointed* if

$$\mathcal{K} \cap (-\mathcal{K}) = \{0\}.$$

It is *proper* if it has nonempty interior in  $\mathcal{S}^n$  and is closed, convex, and pointed.

For a proper cone  $\mathcal{K}$ , we define the *dual cone*  $\mathcal{K}^*$  as

$$\mathcal{K}^* := \{X \in \mathcal{S}^n : \langle Q, Z \rangle \geq 0 \text{ for all } Z \in \mathcal{K}\}.$$

A proper cone  $\mathcal{K}$  is self-dual if  $\mathcal{K}^* = \mathcal{K}$ .

The self-duality of  $\mathcal{S}_+^n$  follows from the following theorem:

**Theorem 1.1.** ([8, Corollary 7.5.4])  $A \geq 0 \iff \langle A, B \rangle \geq 0 \text{ for all } B \geq 0$ .

Hence,

$$\begin{aligned} (\mathcal{S}_+^n)^* &= \{Y \in \mathcal{S}^n : \langle X, Y \rangle \geq 0 \ \forall X \in \mathcal{S}_+^n\} \\ &= \mathcal{S}_+^n \text{ by Theorem 1.1.} \end{aligned}$$

Theorem 1.1 above is only one of the many properties of psd matrices. We present here only a few that are frequently useful; many more facts about psd matrices can be found in [8].

First we have the following well-known theorem about symmetric matrices.

**Theorem 1.2.** ([8, Theorem 4.1.5]) (Spectral theorem)  $A \in \mathcal{S}^n \iff$  there is a real matrix  $U$  such that

$$A = UDU^T$$

with  $U^T U = UU^T = I$ , and  $D$  a real diagonal matrix of the eigenvalues of  $A$ .

**Corollary 1.1.** All the eigenvalues of  $A \in \mathcal{S}^n$  are real.

This leads to a first characterization of psd matrices.

**Theorem 1.3.** ([8, Theorem 7.2.1]) For  $A \in \mathcal{S}^n$ ,

$$A \succeq 0 \iff \text{all the eigenvalues of } A \text{ are non-negative.}$$

**Corollary 1.2.** The trace and determinant of a psd matrix are non-negative.

Another useful characterization of psd matrices is formulated in terms of principal minors.

**Definition 1.4.** Given  $A \in \mathcal{S}^n$  and a subset  $I \subseteq \{1 \dots n\}$ , the principal submatrix of  $A$  corresponding to  $I$  is the submatrix with rows and columns indexed by  $I$ . Its determinant is called the principal minor.

*Example 1.4.* For  $I = \{2, 3\}$  and

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 3 \\ 3 & 3 & 6 \end{pmatrix},$$

the principal submatrix is

$$A(I) = \begin{pmatrix} 4 & 3 \\ 3 & 6 \end{pmatrix}$$

and the principal minor is  $\det A(I) = 15$ .

The corresponding characterization of psd matrices is:

**Theorem 1.4.** ([8, p.405])  $A \succeq 0 \iff$  all the principal minors of  $A$  are non-negative.

Hence, by considering  $I = \{1, 3\}$ , we can prove that the matrix in Example 1.4 is not psd.

A property which is often useful is that every psd matrix has a square root. This follows immediately from Theorems 1.2 and 1.3:

**Theorem 1.5.**  $A \geq 0 \iff \exists \text{ set of vectors } \{w_1, \dots, w_n\} \text{ such that } A_{i,j} = w_i^T w_j, \text{ i.e., } A = W^T W \text{ where } W = (w_1, \dots, w_n).$

Note that the matrix  $W$  is not unique. Indeed, for any orthogonal matrix  $Q \in \mathbb{R}^n$ , let  $W_Q := QW$ . Then  $W_Q^T W_Q = W^T Q^T QW = W^T W = A$ , since  $Q^T Q = I$  by the orthogonality of  $Q$ . A specific choice of  $W$  which is very useful from a computational point of view is the *Cholesky decomposition*:

$$A \geq 0 \iff A = LL^T$$

where  $L$  is a lower triangular matrix. (If  $A$  is pd, then  $L$  is nonsingular with strictly positive entries on the diagonal.) The Cholesky decomposition can be computed efficiently (see for example [5]) and is useful in many practical algorithms for solving SDP problems.

We mention two more important properties of psd matrices.

**Theorem 1.6.** ([8, Theorem 7.7.6]) (*Schur Complement theorem*) If

$$M = \begin{pmatrix} A & B \\ B^T & C \end{pmatrix},$$

and  $A$  is pd, then  $M \geq 0 \iff C - B^T A^{-1} B \geq 0$ .

**Theorem 1.7.** ([8, Theorem 6.1.10]) Let  $A \in \mathcal{S}^n$  have strictly positive diagonal entries, and be strictly diagonally dominant, i.e.,

$$|A_{ii}| > \sum_{j \neq i, j=1}^n |A_{ij}| \quad \text{for } i = 1, \dots, n.$$

Then  $A$  is pd.

### 1.3 Duality in SDP

For (P) and (D) in the primal-dual pair (1.1) defined above, we have (as in LP):

**Theorem 1.8 (Weak Duality).** If  $\tilde{X}$  is feasible for (P) and  $\tilde{y}, \tilde{Z}$  for (D), then  $\langle C, \tilde{X} \rangle - b^T \tilde{y} \leq \langle \tilde{Z}, \tilde{X} \rangle$ .

*Proof.*

$$\langle C, \tilde{X} \rangle - b^T \tilde{y} = \left\langle \left( \sum_{i=1}^m \tilde{y}_i A_i - \tilde{Z} \right), \tilde{X} \right\rangle - \left\langle \sum_i \tilde{y}_i A_i, \tilde{X} \right\rangle = -\langle \tilde{Z}, \tilde{X} \rangle \leq 0,$$

by Theorem 1.1. □

However, because of the nonlinear psd constraint, SDP duality has some cases that do not occur in LP. We illustrate this with the following two examples from [29, Examples 4.1.1 and 4.1.2].

*Example 1.5 (Positive duality gap).* In LP, if both (P) and (D) are feasible, then there is no duality gap. This may fail for SDP. For example,

$$\begin{array}{ll} (\text{P}) \max -ax_{11} & (\text{D}) \min -y_2 \\ \text{s.t. } x_{11} + 2x_{23} = 1 & \\ x_{22} = 0 & \text{s.t. } \begin{pmatrix} y_2 - a & 0 & 0 \\ 0 & y_1 & y_2 \\ 0 & y_2 & 0 \end{pmatrix} \leq 0 \\ X \geq 0. & \end{array}$$

It is easy to see that (P) has optimal objective value  $-a$ , while (D) has 0.

*Example 1.6 (Weak infeasibility).* Even if there is no duality gap at optimality, the optimal value may not be attained for (P) or (D). Consider the primal-dual pair

$$\begin{array}{ll} (\text{P}) \sup 2x_{12} & (\text{D}) \inf y_1 \\ \text{s.t. } \begin{pmatrix} 1 & x_{12} \\ x_{12} & 0 \end{pmatrix} \geq 0 & \text{s.t. } \begin{pmatrix} y_1 & -1 \\ -1 & y_2 \end{pmatrix} \geq 0 \end{array}$$

and observe that the optimal objective value 0 is attained for (P), but not for (D).

We can avoid these difficulties by requiring that the SDP problem and its dual satisfy a constraint qualification. This is a standard technique in nonlinear optimization. The purpose of a constraint qualification is to ensure the existence of Lagrange multipliers at optimality. These multipliers are an optimal solution for the dual problem, and thus the constraint qualification ensures that strong duality holds: it is possible to achieve primal and dual feasibility with no duality gap.

One common choice of constraint qualification is *Slater's constraint qualification*. It is usually easy to verify that it holds for an SDP problem; indeed, it suffices to exhibit a feasible matrix which is pd (psd but not singular) for the SDP problem of interest.

*Example 1.7 (Slater's Constraint Qualification for (1.5) and (1.6)).* For the basic SDP relaxation of max-cut, it is obvious that the  $n \times n$  identity matrix is pd and feasible. Hence, Slater's constraint qualification holds for the SDP problem (1.5).

To verify that it also holds for the dual SDP problem (1.6), choose  $\bar{y}$  sufficiently large so that  $\bar{y}_i > \sum_{j=1}^n Q_{ij}$  for each  $i = 1, \dots, n$ . Then the matrix  $S = \text{Diag}(\bar{y}) - Q$  will be pd (by Theorem 1.7) and feasible for the dual SDP problem (1.6).

## 1.4 Computational Complexity and SDP

The fact that SDP problems can be solved in polynomial-time to within a given accuracy follows from the complexity analysis of the ellipsoid algorithm (see [6]). Specifically, consider the SDP problem (P) defined in (1.1) with integer data, a given rational  $\epsilon > 0$ , and a given integer  $R > 0$  such that either (P) is infeasible or  $\|X\| \leq R$  for some feasible  $X$ . Then it is possible to find in polynomial-time either:

- A matrix  $X^*$  at distance at most  $\epsilon$  from the feasible set of (P) such that  $\langle C, X^* \rangle - p^* \leq \epsilon$ , where  $p^*$  is the optimal value of (P); or
- A certificate that the feasible set of (P) does not contain a ball of radius  $\epsilon$ .

The complexity of the algorithm is polynomial in  $n, m, \log(R), \log(\frac{1}{\epsilon})$ , and the bit length of the input data.

It is worth pointing out that some peculiar situations may occur in SDP that cannot occur in LP. First, there are SDP problems with integer data but no rational optimal solution. For instance, the pair of constraints (taken from [17])

$$\begin{pmatrix} 1 & x \\ x & 2 \end{pmatrix} \succeq 0 \quad \text{and} \quad \begin{pmatrix} 2x & 2 \\ 2 & x \end{pmatrix} \succeq 0$$

have  $x = \sqrt{2}$  as the unique feasible solution (apply Corollary 1.2).

Another situation that may occur in SDP is that all feasible solutions are doubly exponential. Consider the set of constraints (taken from [22])

$$x_1 \geq 2$$

$$\begin{pmatrix} 1 & x_{i-1} \\ x_{i-1} & x_i \end{pmatrix} \succeq 0, i = 2, \dots, n.$$

Then any feasible solution must satisfy  $x_i \geq 2^{2^{i-1}}$ ,  $i = 1, \dots, n$ , which means that every rational feasible solution has exponential bitlength.

For more details on issues of complexity and SDP, we refer the reader to [1, Sect. 1.9] and [17, Sect. 2.3].

## Part II: Polynomial Optimization

### 1.5 Introduction

Consider the optimization problem:

$$\mathbf{P} : \quad f^* := \inf_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{K}\} \quad (1.9)$$

for some given measurable function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and some Borel subset  $\mathbf{K} \subset \mathbb{R}^n$ . Here we insist on the fact that  $f^*$  is the *global* minimum on  $\mathbf{K}$ , as opposed to a *local* minimum. In full generality, problem (1.9) is very difficult and there is no general purpose method to compute (or even approximate)  $f^*$ . However, and fortunately, we will see that in the case where  $f$  is a polynomial and  $\mathbf{K}$  is a basic compact semi-algebraic set, the situation is much better because powerful results from real algebraic geometry can be exploited to develop what we call the *momentos approach* to approximate (and sometimes obtain exactly)  $f^*$ .

### 1.5.1 Two Dual Points of View

Observe that, when one searches for the global minimum  $f^*$ , then

$$f^* = \inf_{\mu \in \mathcal{M}(\mathbf{K})_+} \left\{ \int_{\mathbf{K}} f d\mu : \mu(\mathbf{K}) = 1 \right\}, \quad (1.10)$$

where  $\mathcal{M}(\mathbf{K})_+$  is the space of finite Borel measures on  $\mathbf{K}$  (the convex positive cone of the vector space  $\mathcal{M}(\mathbf{K})$  of finite signed Borel measures on  $\mathbf{K}$ ). Indeed, if  $\mathbf{x} \in \mathbf{K}$  then the Dirac measure  $\mu := \delta_{\mathbf{x}}$  at the point  $\mathbf{x}$  is a feasible solution of (1.10) with value  $\int f d\mu = f(\mathbf{x})$ . Conversely, if  $f \geq f^*$  on  $\mathbf{K}$  and  $\mu$  is feasible for (1.10) then  $\int f d\mu \geq \int f^* d\mu = f^*$ . On the other hand, we also have:

$$f^* = \sup_{\lambda} \{ \lambda : f(\mathbf{x}) - \lambda \geq 0, \quad \forall \mathbf{x} \in \mathbf{K} \}. \quad (1.11)$$

In fact the two formulations (1.10) and (1.11) are dual of each other in the sense of classical LP-duality if one observes that (1.10) is the infinite-dimensional LP

$$f^* = \inf_{\mu \in \mathcal{M}(\mathbf{K})} \{ \langle f, \mu \rangle : \langle 1, \mu \rangle = 1; \quad \mu \geq 0 \} \quad (1.12)$$

where  $\langle g, \mu \rangle = \int_{\mathbf{K}} g d\mu$  for every  $\mu \in \mathcal{M}(\mathbf{K})$  and every bounded measurable function  $g$  on  $\mathbf{K}$  (e.g. assuming that  $f$  is bounded measurable on  $\mathbf{K}$ ).

The equivalent formulation (1.10) shows that (1.9) can be viewed as a particular (and perhaps the simplest) instance of the so-called *Generalized Problem of Moments* (GPM), a very general (and old) problem at the crossroads of several disciplines and with many important potential applications. In fact, the LP-duality between (1.12) and (1.11) illustrates the well-known duality between moments and positive polynomials. For more details the interested reader is referred to e.g. [11, 14] and the many references therein.

Both optimization problems (1.11) and (1.12) are linear programs but infinite-dimensional! In (1.11) one has uncountably many constraints  $f(\mathbf{x}) \geq \lambda$ ,  $\mathbf{x} \in \mathbf{K}$ , whereas in (1.12) the unknown is a signed Borel measure  $\mu \in \mathcal{M}(\mathbf{K})$  (and not a finite dimensional vector as in classical LP). With no other assumption on neither  $f$

nor  $\mathbf{K}$ , one does not know how to solve (1.11) and (1.12) because one does not have *tractable characterizations* of Borel measures supported on  $\mathbf{K} \subset \mathbb{R}^n$  (for solving (1.12)), or functions nonnegative on  $\mathbf{K}$  (for solving (1.11)). And so in general (1.11) and (1.12) are only a *rephrasing* of  $\mathbf{P}$  in (1.9).

However, if one now considers problem (1.9) with the restrictions that:

- (i)  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is a polynomial (or even a rational function),
- (ii)  $\mathbf{K} \subset \mathbb{R}^n$  is a compact basic semi-algebraic set,

then one may solve (or approximate as closely as desired) the linear programs (1.11)–(1.12). Indeed, relatively recent results from real algebraic geometry permit to characterize polynomials that are positive on  $\mathbf{K}$ , which is exactly what we need to solve (1.11). In addition it turns out that those characterizations are tractable as they translate into *semidefinite* (or sometimes *linear*) conditions on the coefficients of certain polynomials that appear in some appropriate representation of the polynomial  $\mathbf{x} \mapsto f(\mathbf{x}) - \lambda$ , positive on  $\mathbf{K}$ . Moreover, the previous representation results have a nice *dual facet* which is concerned with sequences of reals  $\mathbf{y} = (y_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , that are the *moments* of a finite Borel measure  $\mu$  supported on  $\mathbf{K}$ , which is precisely what we need to solve the linear program (1.12) when  $f \in \mathbb{R}[\mathbf{x}]$ .

Even with the above two restrictions (i)–(ii), problem (1.9) still encompasses a lot of important optimization problems. In particular, it includes 0/1 optimization problems, modeling  $x_i \in \{0, 1\}$  via the quadratic polynomial constraint  $x_i^2 = x_i$ . Therefore, one should always have in mind that one addresses NP-hard problems in general.

We next describe results from real algebraic geometry on the representation of polynomials positive on  $\mathbf{K}$ , and their dual counterparts in functional analysis about sequences  $\mathbf{y} = (y_\alpha)$  that have a finite representing Borel measure  $\mu$  supported on  $\mathbf{K}$ . We then show how to use those results to define what we call the *moment-sos* approach. It consists of a hierarchy of *semidefinite relaxations* for problem (1.9), whose associated sequence of optimal values is monotone and converges to the global minimum  $f^*$  (sometimes with even finite convergence). Of course, the higher in the hierarchy the larger the size of the resulting semidefinite program. Another type of representation due to Krivine [9] and not described in this chapter, yields LP relaxations. But those LP-relaxations suffer from serious drawbacks. For more details, the interested reader is referred to e.g. [12, 14].

An important feature of the moment-sos approach is to *not* distinguish between convex continuous problems (considered easy) and non convex (possibly discrete) problems (considered hard). For instance, a boolean variable  $x_i$  is not treated with any particular attention and is just modeled via the quadratic equality constraint  $x_i^2 - x_i = 0$  in the definition of the feasible set  $\mathbf{K}$ . This might justify some doubts concerning the efficiency of the moment-sos approach by lack of specialization. Yet, and remarkably, the resulting semidefinite relaxations still provide the strongest relaxation algorithms for hard combinatorial optimization problems [16] and in the same time they also recognize easy convex problems as in this case convergence is even finite (and sometimes at the first semidefinite relaxation) [14]!

At last but not least, the moment-sos approach has also been shown to be successful in a large variety of applications and not only in optimization. In fact it applies to any problem whose description involves either linear constraints on finite Borel measures or, dually, positivity constraints of polynomials on basic semi-algebraic sets, or both (like in games)! Some of such applications are described in e.g. [10, 14, 18, 19].

## 1.6 Moments and Positive Polynomials

Let  $\mathbb{R}[\mathbf{x}]$  be the ring of polynomials in the variables  $\mathbf{x} = (x_1, \dots, x_n)$ . Denote by  $\mathbb{R}[\mathbf{x}]_d \subset \mathbb{R}[\mathbf{x}]$  the vector space of polynomials of degree at most  $d$ , which forms a vector space of dimension  $s(d) = \binom{n+d}{d}$ , with e.g., the usual canonical basis  $(\mathbf{x}^\alpha)$  of monomials. Also, denote by  $\Sigma[\mathbf{x}] \subset \mathbb{R}[\mathbf{x}]$  (resp.  $\Sigma[\mathbf{x}]_d \subset \mathbb{R}[\mathbf{x}]_{2d}$ ) the space of sums of squares (s.o.s.) polynomials (resp. s.o.s. polynomials of degree at most  $2d$ ). If  $f \in \mathbb{R}[\mathbf{x}]_d$ , write  $f(\mathbf{x}) = \sum_{\alpha \in \mathbb{N}^n} f_\alpha \mathbf{x}^\alpha$  in the canonical basis and denote by  $\mathbf{f} = (f_\alpha) \in \mathbb{R}^{s(d)}$  its vector of coefficients. Finally, recall that  $\mathcal{S}^n$  denotes the space of  $n \times n$  real symmetric matrices, with inner product  $\langle \mathbf{A}, \mathbf{B} \rangle = \text{trace } \mathbf{AB}$ , and where the notation  $\mathbf{A} \succeq 0$  (resp.  $\mathbf{A} > 0$ ) stands for  $\mathbf{A}$  is positive semidefinite (resp. positive definite).

A sequence  $\mathbf{y} = (y_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , has a representing measure  $\mu$  on  $\mathbb{R}^n$  if

$$y_\alpha = \int_{\mathbb{R}^n} \mathbf{x}^\alpha d\mu(\mathbf{x}), \quad \forall \alpha \in \mathbb{N}^n,$$

for some finite Borel measure  $\mu \in \mathcal{M}(\mathbf{K})_+$ , and  $y_\alpha$  is called the  $\alpha$ -moment of  $\mu$ . The measure  $\mu$  is said to be *determinate* if there is no other measure with same moments. (On a compact set, every measure is determinate.)

### 1.6.1 Moment Matrix

With a sequence  $\mathbf{y} = (y_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , let  $L_y : \mathbb{R}[\mathbf{x}] \rightarrow \mathbb{R}$  be the linear functional

$$h \quad \left(= \sum_{\alpha} h_\alpha \mathbf{x}^\alpha\right) \quad \mapsto \quad L_y(h) = \sum_{\alpha} h_\alpha y_\alpha, \quad h \in \mathbb{R}[\mathbf{x}].$$

With  $d \in \mathbb{N}$ , let  $\mathbb{N}_d^n := \{\alpha \in \mathbb{N}^n : \sum_i \alpha_i \leq d\}$ , and let  $\mathbf{M}_d(\mathbf{y})$  be the real symmetric matrix with rows and columns indexed in  $\mathbb{N}_d^n$ , and defined by:

$$\mathbf{M}_d(\mathbf{y})(\alpha, \beta) := L_y(\mathbf{x}^{\alpha+\beta}) = y_{\alpha+\beta}, \quad \alpha, \beta \in \mathbb{N}_d^n. \quad (1.13)$$

The matrix  $\mathbf{M}_d(\mathbf{y})$  is called the moment matrix associated with  $\mathbf{y}$  and

$$\langle \mathbf{g}, \mathbf{M}_d(\mathbf{y}) \mathbf{g} \rangle = L_y(g^2), \quad \forall g \in \mathbb{R}[\mathbf{x}]_d, \quad (1.14)$$

so that

$$\left[ L_{\mathbf{y}}(g^2) \geq 0 \quad \forall g \in \mathbb{R}[\mathbf{x}] \right] \Leftrightarrow \mathbf{M}_d(\mathbf{y}) \succeq 0, \quad d = 0, 1, \dots$$

### 1.6.2 Localizing Matrix

Similarly, with  $\mathbf{y} = (y_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , and  $f \in \mathbb{R}[\mathbf{x}]$  written

$$\mathbf{x} \mapsto f(\mathbf{x}) = \sum_{\gamma \in \mathbb{N}^n} f_\gamma \mathbf{x}^\gamma,$$

let  $\mathbf{M}_d(f\mathbf{y})$  be the real symmetric matrix with rows and columns indexed in  $\mathbb{N}_d^n$ , and defined by:

$$\mathbf{M}_d(f\mathbf{y})(\alpha, \beta) := L_{\mathbf{y}}(f(\mathbf{x}) \mathbf{x}^{\alpha+\beta}) = \sum_{\gamma} f_\gamma y_{\alpha+\beta+\gamma}, \quad \forall \alpha, \beta \in \mathbb{N}_d^n. \quad (1.15)$$

The matrix  $\mathbf{M}_d(f\mathbf{y})$  is called the localizing matrix associated with  $\mathbf{y}$  and  $f \in \mathbb{R}[\mathbf{x}]$ . Actually, the localizing matrix  $\mathbf{M}_d(f\mathbf{y})$  is nothing less than the moment matrix associated with the sequence  $\mathbf{z} = f\mathbf{y} = (z_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , with  $z_\alpha = \sum_\gamma f_\gamma y_{\alpha+\gamma}$ . Observe that

$$\langle \mathbf{g}, \mathbf{M}_d(f\mathbf{y}) \mathbf{g} \rangle = L_{\mathbf{y}}(g^2 f), \quad \forall g \in \mathbb{R}[\mathbf{x}]_d, \quad (1.16)$$

and so

$$\left[ L_{\mathbf{y}}(g^2 f) \geq 0 \quad \forall g \in \mathbb{R}[\mathbf{x}] \right] \Leftrightarrow \mathbf{M}_d(f\mathbf{y}) \succeq 0, \quad d = 0, 1, \dots$$

If  $\mathbf{y}$  has a representing measure  $\mu$  then (1.16) reads

$$\langle \mathbf{g}, \mathbf{M}_d(f\mathbf{y}) \mathbf{g} \rangle = L_{\mathbf{y}}(g^2 f) = \int g(\mathbf{x})^2 f(\mathbf{x}) d\mu(\mathbf{x}), \quad \forall g \in \mathbb{R}[\mathbf{x}]_d, \quad (1.17)$$

and if  $\mu$  has its support contained in the level set  $\{\mathbf{x} : f(\mathbf{x}) \geq 0\}$ , then  $\mathbf{M}_d(f\mathbf{y}) \succeq 0$  for all  $d = 0, 1, \dots$

### 1.6.3 Certificates of Positivity

Let  $\mathbf{K} \subset \mathbb{R}^n$  be the basic semi-algebraic set

$$\mathbf{K} := \{\mathbf{x} \in \mathbb{R}^n : g_j(\mathbf{x}) \geq 0, \quad j = 1, \dots, m\}, \quad (1.18)$$

for some polynomials  $(g_j) \subset \mathbb{R}[\mathbf{x}]$ , and with  $g_0$  being the constant polynomial 1, define the following convex cones of  $\mathbb{R}[\mathbf{x}]$ :

$$P(g) := \left\{ \sum_{J \subseteq \{1, \dots, m\}} \sigma_J \left( \prod_{k \in J} g_k \right) : \sigma_J \in \Sigma[\mathbf{x}] \right\}, \quad (1.19)$$

$$Q(g) := \left\{ \sum_{j=0}^m \sigma_j g_j : \sigma_j \in \Sigma[\mathbf{x}] \right\}, \quad (1.20)$$

with the convention that if  $J = \emptyset$  then  $\prod_{k \in \emptyset} g_k = 1$ . The set  $P(g)$  (resp.  $Q(g)$ ) is called the *preordering* (resp. *quadratic module*) associated with the polynomials  $(g_j)$ . Of course, when  $f \in P(g)$  (resp.  $Q(g)$ ), the elements  $(\sigma_J)$  in (1.19) (resp.  $(\sigma_j)$  in (1.20)) provide an obvious certificate of nonnegativity for  $f$  on  $\mathbf{K}$ . Notice that if on the one hand the cones  $P(g)$  and  $Q(g)$  are associated with  $\mathbf{K}$ , on the other hand they depend on the  $g_j$ 's, i.e., on the *representation* of  $\mathbf{K}$ .

**Theorem 1.9 (Stengle's Nichtnegativstellensatz).** *Let  $\mathbf{K} \subseteq \mathbb{R}^n$  be as in (1.18) (not necessarily compact). A polynomial  $f \in \mathbb{R}[\mathbf{x}]$  is nonnegative on  $\mathbf{K}$  if and only if  $hf = f^{2t} + g$  for some  $t \in \mathbb{N}$ , and some  $h, g \in P(g)$ . Moreover there are upper bounds available for the exponent  $t$  and the degree of the s.o.s. weights  $\sigma_J$  in the definition of the elements  $h$  and  $g$  of  $P(g)$ .*

Theorem 1.9 is extremely powerful. For instance checking whether  $f \geq 0$ , i.e., whether  $hf = f^{2t} + g$  for some  $h, g \in P(g)$ , reduces to solve a single semidefinite program. However, the degree bounds are unpractical and therefore the size of the resulting semidefinite program is out of reach. On the other hand, one still may try to find a certificate  $t, h, g$  with relatively small degree bounds fixed, a priori. The next result is more practical and assumes that  $\mathbf{K}$  in (1.18) is compact.

**Theorem 1.10 (Schmüdgen's Positivstellensatz [23]).** *Let  $\mathbf{K} \subset \mathbb{R}^n$  in (1.18) be compact. Then:*

(a) *A polynomial  $f \in \mathbb{R}[\mathbf{x}]$  is strictly positive on  $\mathbf{K}$  only if  $f \in P(g)$ , i.e.,*

$$f = \sum_{J \subseteq \{1, \dots, m\}} \sigma_J \left( \prod_{k \in J} g_k \right),$$

*for some s.o.s. polynomials  $(\sigma_J) \subset \Sigma[\mathbf{x}]$ . Moreover there are upper bounds available for the degree of the s.o.s. weights  $\sigma_J$ .*

(b) *A sequence  $\mathbf{y} = (y_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , has a representing measure on  $\mathbf{K}$  if and only if:*

$$\mathbf{M}_d(g_J \mathbf{y}) \geq 0, \quad \forall J \subseteq \{1, \dots, m\}, \quad (1.21)$$

*where  $g_J := \prod_{k \in J} g_k$ .*

Theorem 1.10(a) is the real algebraic geometry facet whereas Theorem 1.10(b) is the functional analysis (dual) facet. Notice that there is no multiplier of  $f$  as in

**Theorem 1.9.** On the other hand,  $\mathbf{K}$  is assumed to be compact and  $f$  is strictly positive on  $\mathbf{K}$ . But the certificate still has  $2^m$  unknown weights! The next result is a refinement which avoids computing  $2^m$  s.o.s. weights.

**Assumption 1.1** *There is some  $N > 0$  such that the quadratic polynomial  $\mathbf{x} \mapsto N - \|\mathbf{x}\|^2$  belongs to  $Q(g)$ , i.e., it can be written in the form*

$$N - \|\mathbf{x}\|^2 = \sigma_0 + \sum_{k=1}^m \sigma_j g_j,$$

for some s.o.s. polynomials  $(\sigma_j)_{j=1}^m \subset \Sigma[\mathbf{x}]$ .

Observe that under Assumption 1.1,  $\mathbf{K}$  is necessarily compact. On the other hand, if  $\mathbf{K}$  is compact then  $N - \|\mathbf{x}\|^2 \geq 0$  on  $\mathbf{K}$  provided that  $N$  is large enough, and so by including the redundant quadratic constraint  $g_{m+1}(\mathbf{x}) := N - \|\mathbf{x}\|^2 \geq 0$  in the definition of  $\mathbf{K}$ , Assumption 1.1 holds automatically.

**Theorem 1.11 (Putinar's Positivstellensatz [21]).** *Let  $\mathbf{K}$  be as in (1.18) and let Assumption 1.1 hold. Then:*

(a) *A polynomial  $f \in \mathbb{R}[\mathbf{x}]$  is strictly positive on  $\mathbf{K}$  only if  $f \in Q(g)$ , i.e.,*

$$f = \sigma_0 + \sum_{j=1}^m \sigma_j g_j, \quad (1.22)$$

for some s.o.s. polynomials  $(\sigma_j) \subset \Sigma[\mathbf{x}]$ . Moreover there are upper bounds available for the degree of the s.o.s. weights  $\sigma_j$ .

(b) *A sequence  $\mathbf{y} = (y_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , has a representing measure on  $\mathbf{K}$  if and only if:*

$$\mathbf{M}_d(\mathbf{y}) \succeq 0; \quad \mathbf{M}_d(g_j \mathbf{y}) \succeq 0, \quad j = 1, \dots, m, \quad (1.23)$$

for all  $d = 0, 1, \dots$

Again, Theorem 1.11(a) is the real algebraic geometry facet whereas Theorem 1.11(b) is the functional analysis (dual) facet. Both Theorem 1.10 and 1.11 are results of primary importance which provide *certificates* of positivity of  $f$  on  $\mathbf{K}$ . However, Theorem 1.11 is more appealing because it only requires  $m+1$  s.o.s. weights (as opposed to  $2^m$  in Theorem 1.10). As already mentioned, the price to pay for this refinement (Assumption 1.1 has to hold) is negligible as one may always include a redundant quadratic constraint so as to make Assumption 1.1 hold.

In addition of being of self-interest, the importance of Theorem 1.11 stems from the fact that it is amenable to practical computation. Indeed:

- Given  $f \in \mathbb{R}[\mathbf{x}]$  and an a priori degree bound on the s.o.s. weights  $\sigma_j$ , checking whether (1.22)(a) holds reduces to solving a semidefinite program.
- Given a sequence  $\mathbf{y}$ , and  $d \in \mathbb{N}$  fixed, checking whether (1.23) holds reduces to check whether  $m+1$  LMIs are satisfied.

These two properties provide a rationale for the *moment-sos* approach described in the next section, which builds up a hierarchy of semidefinite relaxations to compute (or at least approximate)  $f^*$  in (1.9).

## 1.7 A Hierarchy of Semidefinite Relaxations

Consider the global optimization problem (1.9) with  $\mathbf{K}$  as in (1.18), and let  $v_j := \lceil (\deg g_j)/2 \rceil$ ,  $j = 1, \dots, m$ , and let  $v_0 = 0$ . For  $d \geq \max_j v_j$ , consider the semidefinite program  $\mathbf{Q}_d$  defined by:

$$\mathbf{Q}_d : \left\{ \begin{array}{ll} \rho_d = \min_{\mathbf{y}} & L_{\mathbf{y}}(f) \\ \text{s.t.} & \mathbf{M}_d(\mathbf{y}) \geq 0 \\ & \mathbf{M}_{d-v_j}(g_j \mathbf{y}) \geq 0, \quad j = 1, \dots, m \\ & y_0 = 1. \end{array} \right. \quad (1.24)$$

Obviously,  $\mathbf{Q}_d$  is a relaxation of problem (1.10). Indeed, with  $\mathbf{x} \in \mathbf{K}$  arbitrary, let  $\mathbf{y} := (\mathbf{x}^\alpha)$ ,  $\alpha \in \mathbb{N}_{2d}^n$ . Then  $\mathbf{y}$  is feasible for (1.24) with value  $L_{\mathbf{y}}(f) = f(\mathbf{x})$ . Therefore,  $\rho_d \leq f^*$ . The conditions  $\mathbf{M}_d(\mathbf{y}), \mathbf{M}_{d-v_j}(g_j \mathbf{y}) \geq 0$  are necessary conditions for  $\mathbf{y}$  to be the vector of moments of a measure  $\mu$  (a probability measure as  $y_0 = 1$ ) supported on  $\mathbf{K}$ . The sequence of semidefinite programs  $(\mathbf{Q}_d)$ ,  $d \in \mathbb{N}$ , forms a *hierarchy* because the associated feasible sets form a nested sequence.

Ideally, if  $\mathbf{x}^* \in \mathbf{K}$  is a maximizer, one would like an optimal solution of (1.24) to be the vector of moments associated with the Dirac measure  $\mu^* := \delta_{\mathbf{x}^*}$  at  $\mathbf{x}^*$ , because  $\mu^*$  is an optimal solution of (1.10).

If  $\mathbf{v}_d(\mathbf{x})$  is the vector  $(\mathbf{x}^\alpha)$ ,  $\alpha \in \mathbb{N}_d^n$ , then

$$\mathbf{v}_d(\mathbf{x})\mathbf{v}_d(\mathbf{x})^T = \sum_{\alpha \in \mathbb{N}_{2d}^n} \mathbf{x}^\alpha \mathbf{B}_\alpha; \quad g_j(\mathbf{x})\mathbf{v}_{d-v_j}(\mathbf{x})\mathbf{v}_{d-v_j}(\mathbf{x})^T = \sum_{\alpha \in \mathbb{N}_{2d}^n} \mathbf{x}^\alpha \mathbf{C}_\alpha^j,$$

for appropriate real symmetric matrices  $\mathbf{B}_\alpha$  and  $\mathbf{C}_\alpha^j$ ,  $j = 1, \dots, m$ . Hence,

$$\mathbf{M}_d(\mathbf{y}) = \sum_{\alpha \in \mathbb{N}_{2d}^n} y_\alpha \mathbf{B}_\alpha, \quad \mathbf{M}_{d-v_j}(g_j \mathbf{y}) = \sum_{\alpha \in \mathbb{N}_{2d}^n} y_\alpha \mathbf{C}_\alpha^j, \quad j = 1, \dots, m,$$

and the dual of (1.24) is the semidefinite program  $\mathbf{Q}_d^*$  defined by:

$$\left\{ \begin{array}{ll} \rho_d^* = \max_{\lambda, \mathbf{Z}_j} & \lambda \\ \text{s.t.} & f_\alpha - \lambda \delta_{\alpha=0} = \langle \mathbf{Z}_0, \mathbf{B}_\alpha \rangle + \sum_{j=1}^m \langle \mathbf{Z}_j, \mathbf{C}_\alpha^j \rangle, \quad \forall \alpha \in \mathbb{N}_{2d}^n \\ & \mathbf{Z}_j \succeq 0, \quad j = 0, 1, \dots, m \end{array} \right. \quad (1.25)$$

(where  $\delta$  is the Kronecker symbol). Observe that, given a feasible solution  $(\lambda, (\mathbf{Z}_j))$ , multiplying both sides of the constraint in (1.25) by  $\mathbf{x}^\alpha$  and summing up, yields

$$f(\mathbf{x}) - \lambda = \langle \mathbf{Z}_0, \mathbf{v}_d(\mathbf{x})\mathbf{v}_d(\mathbf{x})^T \rangle + \sum_{j=1}^m g_j(\mathbf{x})\langle \mathbf{Z}_j, \mathbf{v}_{d-v_j}(\mathbf{x})\mathbf{v}_{d-v_j}(\mathbf{x})^T \rangle.$$

For every  $j = 0, 1, \dots, m$ , write  $\mathbf{Z}_j = \sum_k \mathbf{q}_{jk} \mathbf{q}_{jk}^T$  for some vectors  $(\mathbf{q}_{jk})$ . Then

$$\langle \mathbf{Z}_0, \mathbf{v}_d(\mathbf{x})\mathbf{v}_d(\mathbf{x})^T \rangle = \sum_k (\mathbf{q}_{0k}^T \mathbf{v}_d(\mathbf{x}))^2 = \sum_k q_{0k}(\mathbf{x})^2 (= \sigma_0(\mathbf{x}) \in \Sigma[\mathbf{x}]_d)$$

for some polynomials  $(q_{0k}) \subset \mathbb{R}[\mathbf{x}]_d$  with coefficient vectors  $(\mathbf{q}_{0k})$ . Similarly, for every  $j = 1, \dots, m$ ,

$$\begin{aligned} g_j(\mathbf{x})\langle \mathbf{Z}_j, \mathbf{v}_{d-v_j}(\mathbf{x})\mathbf{v}_{d-v_j}(\mathbf{x})^T \rangle &= g_j(\mathbf{x}) \sum_k (\mathbf{q}_{jk}^T \mathbf{v}_d(\mathbf{x}))^2 = g_j(\mathbf{x}) \sum_k q_{jk}(\mathbf{x})^2 \\ &= g_j(\mathbf{x})\sigma_j(\mathbf{x}) \quad (\sigma_j \in \Sigma[\mathbf{x}]_{d-v_j}), \end{aligned}$$

for some polynomials  $(q_{jk}) \subset \mathbb{R}[\mathbf{x}]_{d-v_j}$  with coefficient vectors  $(\mathbf{q}_{jk})$ . Hence (with  $v_0 := 0$ ) the dual (1.25) has the equivalent formulation

$$\left\{ \begin{array}{l} \rho_d^* = \max_{\lambda, \mathbf{Z}_j} \lambda \\ \text{s.t. } f - \lambda = \sigma_0 + \sum_{j=1}^m \sigma_j g_j \\ \sigma_j \in \Sigma[\mathbf{x}]_{d-v_j}, \quad j = 0, 1, \dots, m. \end{array} \right. \quad (1.26)$$

Observe that if (1.24) is a relaxation of (1.10), by duality (1.26) is a *strengthening* of (1.11) where the (hard) constraint  $f - \lambda \geq 0$  is replaced by the (more tractable) constraint  $f - \lambda \in Q(g)$  (with degree bound on the weights  $\sigma_j$ ).

Let  $v := \max_j v_j$ .

**Theorem 1.12.** *Let  $\mathbf{K}$  be as in (1.18), let Assumption 1.1 hold and consider the hierarchy of semidefinite programs (1.24)–(1.26).*

- (a)  $\rho_d^* \leq \rho_d \leq f^*$  for every  $d \in \mathbb{N}$  and  $\rho_d^*, \rho_d \uparrow f^*$  as  $d \rightarrow \infty$ .
- (b) If  $\mathbf{x}^* \in \mathbf{K}$  is the unique global minimizer of  $\mathbf{P}$  and  $\mathbf{y}^d$  is an optimal (or near optimal solution) of (1.24) then  $\lim_{d \rightarrow \infty} L_{\mathbf{y}^d}(x_i) = \mathbf{x}_i^*$  for every  $i = 1, \dots, n$ .
- (c) Let  $\mathbf{y}^d$  be an optimal solution of (1.24). If

$$\operatorname{rank} \mathbf{M}_d(\mathbf{y}^d) = \operatorname{rank} \mathbf{M}_{d-v}(\mathbf{y}^d) (=: s), \quad (1.27)$$

then  $\rho_d = f^*$  and one may extract  $s$  global minimizers  $\mathbf{x}(k) \in \mathbf{K}$  of  $\mathbf{P}$ .

For a proof the interested reader is referred to e.g. [11, 14, 24]. When (1.27) holds, extracting the  $s$  global minimizers can be done by a linear algebra procedure described in e.g. [14] and implemented in the software package GloptiPoly [7].

So Theorem 1.12 states that in the general context of polynomial optimization, one may approximate as closely as desired the optimal value  $f^*$  and sometimes obtain exactly the global minimizers of  $\mathbf{P}$ , by using the moment-sos approach.

### 1.7.1 Nonlinear 0/1 Programs

For nonlinear 0/1 programs, the feasible set  $\mathbf{K} \subseteq \{0, 1\}^n$  is of the form

$$\mathbf{K} = \{\mathbf{x} \in \mathbb{R}^n : g_j(\mathbf{x}) \geq 0, j = 1, \dots, m; \quad x_i^2 = x_i, i = 1, \dots, n\}, \quad (1.28)$$

which yields simplifications in the moment and localizing matrices. Indeed, in (1.24) the boolean constraints  $x_i^2 = x_i$  imply  $y_\alpha = y_\beta$  where for every  $i, \beta_i = 1$  if  $\alpha_i > 0$  and 0 otherwise. And in fact, in the moment and localizing matrices  $\mathbf{M}_d(\mathbf{y}), \mathbf{M}_d(g_j \mathbf{y})$ , one only retains the rows and columns indexed by the monomials  $\mathbf{x}^\alpha$ ,  $\alpha \in \mathbb{N}_d^n$ , such that  $\alpha_i \leq 1$  for all  $i$ . This is because a column associated with  $\alpha \in \mathbb{N}_d^n$  is identical to the column  $\beta$  where  $\beta_k = 1$ , if  $\alpha_k > 0$ , and  $\beta_k = 0$  otherwise.

For such 0/1 programs, the convergence  $\rho_d \uparrow f^*$  in Theorem 1.12(a) is finite, i.e., there is some  $d \in \mathbb{N}$  such that  $\rho_d = f^*$ . It was also proved in [16] that the semidefinite relaxations  $(\mathbf{Q}_d)$  in (1.24) are the tightest when compared to the LP relaxations (RLT) of Sherali–Adams and the semidefinite lift-and-project relaxations of Lovász–Schrijver. In particular, the hierarchy of semidefinite relaxations  $(\mathbf{Q}_d)$  has attracted a lot of attention from computer scientists to prove (or disprove) inapproximability results for some well-known hard combinatorial problems.

Observe that the moment-sos approach *does not* distinguish between convex continuous problems (considered easy) and non convex (possibly discrete) problems (considered hard). Indeed, the boolean constraint  $x_i^2 - x_i = 0$  is just considered one among the semi-algebraic constraints that define the feasible set  $\mathbf{K}$ , and is not treated with any particular attention. This might justify some doubts concerning the efficiency of the moment-sos approach by lack of specialization. After all, so far, a common feature of all discrete optimization methods is precisely to avoid representing boolean variables by the quadratic equality constraint “ $x_i^2 - x_i = 0$ ” as we do in (1.28). And indeed, using the quadratic constraint “ $x_i^2 - x_i = 0$ ” in most algorithms of continuous optimization in the literature is well-known to be not a good idea! Yet, and remarkably, the semidefinite relaxations (1.24) still provide the strongest relaxation algorithms for hard combinatorial optimization problems [16] and in the same time they also *recognize* easy convex problems as in this case convergence is even finite (and sometimes occurs at the first semidefinite relaxation of the hierarchy)! see e.g. [15].

### 1.7.2 Handling Large Size Problems

From computational experiments, it seems that the convergence  $\rho_d \uparrow f^*$  is fast and even finite in many cases. However, the size of the semidefinite program (1.24) grows like  $O(n^d)$  with  $n$  (when  $d$  is fixed) and like  $O(d^n)$  with  $d$  (when  $n$  is fixed). Therefore, in view of the present status of semidefinite solvers, this makes the moment-sos approach realistic only for small to medium size problems.

Fortunately, very often, large size problems exhibit some symmetries or some sparsity pattern which, when exploited, result in appropriate tractable semidefinite relaxations of much smaller size. For results in this direction, the interested reader is referred to e.g. the works of [2, 3, 25] for symmetries, and [13, 26] for sparsity. For instance, in [26] the authors could solve some non convex polynomial optimization problems with up to a thousand variables!

### 1.7.3 Software

To implement the semidefinite relaxations (1.24)–(1.26), a number of user-friendly public software packages are available on internet. Among them let us cite GloptiPoly [7] dedicated to solving the Generalized Problem of Moments (whose global optimization is only a particular case), SOSTOOLS [20] for general s.o.s. programming, and SparsePOP [27] which is dedicated to solving large scale polynomial optimization problems with structured sparsity.

**Acknowledgements** The first author gratefully acknowledges the support provided by the Alexander von Humboldt Foundation and the Natural Sciences and Engineering Research Council of Canada.

## References

1. de Klerk, E.: Aspects of Semidefinite Programming. Kluwer Academic Publishers, Dordrecht, (2002)
2. de Klerk, E., Pasechnik, D.V., Schrijver, A.: Reduction of symmetric semidefinite programs using the regular  $\star$ -representation. *Math. Program.* **109**, 613–624 (2007)
3. Gaterman, K., Parrilo, P.: Symmetry group, semidefinite programs and sums of squares. *J. Pure Appl. Alg.* **192**, 95–128 (2004)
4. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.* **42**(6), 1115–1145 (1995)
5. Golub, G.H., Van Loan, C.F.: Matrix Computations. Johns Hopkins University Press, Baltimore, MD, third edition (1996)
6. Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Springer-Verlag, Berlin, second edition (1993)

7. Henrion, D., Lasserre, J.B., Lofberg, Y.: Gloptipoly 3: moments, optimization and semidefinite programming. *Optim. Methods and Software* **24**, 761–779 (2009)
8. Horn, R.A., Johnson, C.R.: *Matrix Analysis*. Cambridge University Press, Cambridge (1990)
9. Krivine, J.L.: Anneaux préordonnés. *J. Anal. Math.* **12**, 307–326 (1964)
10. Laraki, R., Lasserre, J.B.: Semidefinite programming for min-max problems and games. *Math. Program.* 1–28 <http://dx.doi.org/10.1007/s10107-010-0353-y>
11. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. *SIAM J. Optim.* **11**, 796–817 (2001)
12. Lasserre, J.B.: Polynomial programming: LP-relaxations also converge. *SIAM J. Optim.* **15**, 383–393 (2004)
13. Lasserre, J.B.: Convergent SDP-relaxations in polynomial optimization with sparsity. *SIAM J. Optim.* **17**, 822–843 (2006)
14. Lasserre, J.B.: *Moments, Positive Polynomials and Their Applications*. Imperial College Press, London (2009)
15. Lasserre, J.B.: Convexity in semi-algebraic geometry and polynomial optimization. *SIAM J. Optim.* **19**, 1995–2014 (2009)
16. Laurent, M.: A comparison of the Sherali-Adams, Lovász-Schrijver and Lasserre relaxations for 0-1 programming. *Math. Oper. Res.* **28**, 470–496 (2003)
17. Laurent, M., Rendl, F.: Semidefinite programming and integer programming. In: Aardal, K., Nemhauser, G., Weismantel, R. (eds.) *Handbook on Discrete Optimization*, Elsevier (2005)
18. Parrilo, P.A.: Semidefinite programming relaxations for semialgebraic problems. *Math. Program. Ser. B* **96**, 293–320 (2003)
19. Parrilo, P.A.: Polynomial games and sum of squares optimization. Proceedings of the 45th IEEE Conference on Decision and Control, San Diego, pp. 2855–2860 (2006)
20. Prajna, S., Papachristodoulou A., Parrilo, P.: Introducing SOSTOOLS: a general purpose sum of squares programming solver. Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, pp. 741–746 (2002)
21. Putinar, M.: Positive polynomials on compact semi-algebraic sets. *Indiana Univ. Math. J.* **42**, 969–984 (1993)
22. Ramana, M.V.: An exact duality theory for semidefinite programming and its complexity implications. *Math. Program.*, 77(2, Ser. B), 129–162 (1997)
23. Schmüdgen, K.: The  $K$ -moment problem for compact semi-algebraic sets. *Math. Ann.* **289**, 203–206 (1991)
24. Schweighofer, M.: Optimization of polynomials on compact semialgebraic sets. *SIAM J. Optim.* **15**, 805–825 (2005)
25. Vallentin, F.: Symmetry in semidefinite programs. *Linear Alg. Appl.* **430**, 360–369 (2009)
26. Waki, S., Kim, S., Kojima, M., Maramatsu, M.: Sums of squares and semidefinite programming relaxations for polynomial optimization problems with structured sparsity. *SIAM J. Optim.* **17**, 218–242 (2006)
27. Waki, H., Kim, S., Kojima, M., Muramatsu, M., Sugimoto, H.: SparsePOP: a Sparse Semidefinite Programming Relaxation of Polynomial Optimization Problems. *ACM Trans. Math. Software* **35**(2), 15:1–15:13 (2008)
28. Wolkowicz, H., Anjos, M.F.: Semidefinite programming for discrete optimization and matrix completion problems. *Discrete Appl. Math.*, 123(1–2), 513–577 (2002)
29. Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.): *Handbook of Semidefinite Programming*. Kluwer Academic Publishers, Boston, MA (2000)

# **Part I**

# **Theory**

# Chapter 2

## The Approach of Moments for Polynomial Equations

Monique Laurent and Philipp Rostalski

### 2.1 Introduction

Computing all points  $x \in \mathbb{K}^n$  ( $\mathbb{K} = \mathbb{R}$  or  $\mathbb{C}$ ) at which a given system of polynomials in  $n$  variables

$$h_1, \dots, h_m \in \mathbb{R}[x_1, \dots, x_n] = \mathbb{R}[x]$$

vanishes simultaneously, is an old problem arising in many mathematical models in science and engineering, with numerous applications in different areas ranging from control, cryptography, computational geometry, coding theory and computational biology to optimization, robotics, statistics and many others (see, e.g., [43]). In this chapter we will focus on the characterization and the (numerical) computation of all real roots or, more generally, of all roots lying in some given basic semi-algebraic set, i.e., satisfying some prescribed polynomial inequalities. A variety of methods has been proposed to tackle such problems, some of which will be briefly recalled in the next section. In this chapter we will focus on a new approach based on sums of squares of polynomials and the dual theory of moments. In this context, semidefinite programming will be the tool permitting to distinguish algorithmically between *real* and *complex nonreal* elements.

---

M. Laurent (✉)

Centrum voor Wiskunde en Informatica, Science Park 123, 1098 XG Amsterdam

Tilburg University, PO Box 90153, 5000 LE Tilburg, The Netherlands

e-mail: [M.Laurent@cwi.nl](mailto:M.Laurent@cwi.nl)

P. Rostalski

Institut für Mathematik, Johann Wolfgang Goethe-Universität,  
Robert-Mayer-Str. 6-10, 60325 Frankfurt am Main, Germany

e-mail: [rostalsk@math.uni-frankfurt.de](mailto:rostalsk@math.uni-frankfurt.de)

## 2.1.1 Existing Methods

Solving polynomial equations has a long tradition covered in a vast literature; for information and further references see e.g. the monographs of Basu et al. [2], Dickenstein and Emiris [9], Mora [27, 28], Elkadi and Mourrain [10], Stetter [42], Sturmels [43]. We do not attempt a complete description of all existing methods, but instead we only try to give a coarse classification. Most existing algorithms can be roughly categorized according to the following criteria: local vs. global search, numerical vs. exact/symbolic computation, and solving over the complex numbers vs. solving over the real numbers.

### 2.1.1.1 Existing Methods over the Complex Numbers

#### Symbolic Methods

Gröbner bases, resultants or, more generally, border bases and generalized normal form algorithms are typical representatives of this class of methods. The main idea is to compute the structure of the quotient algebra  $\mathbb{R}[x]/I$  (where  $I$  is the ideal generated by the given polynomials  $h_i$ ) and to use this information to characterize the roots, e.g., using the shape lemma, or Stickelberger's theorem (viz. the eigenvalue method), or the rational univariate representation.

The following basic fact plays a crucial role: The system of polynomial equations  $h_1 = \dots = h_m = 0$  has finitely many roots if and only if the quotient ring  $\mathbb{R}[x]/I$  of the underlying ideal  $I = \langle h_1, \dots, h_m \rangle$  is finite dimensional as a vector space. This in turn enables to *reduce the computation of all complex roots to tasks of finite dimensional linear algebra* (like eigenvalue computations). Roughly speaking, the basic idea is to replace the given system  $h_i = 0$  by a new equivalent system  $g_j = 0$  with the same set of complex roots, but with a much easier structure facilitating the extraction of the roots.

For instance, one may find an equivalent system comprising polynomials in triangular form  $g_1 \in \mathbb{R}[x_1], g_2 \in \mathbb{R}[x_1, x_2], \dots, g_n \in \mathbb{R}[x_1, \dots, x_n]$ , which can be solved by solving a sequence of *univariate* root finding problems. Such an approach suffers however from the propagation of numerical errors and triangular representations are difficult to compute, typically involving lexicographic Gröbner bases. A more efficient approach is the rational univariate representation, where the new system has a parametric representation:

$$x_1 = h_1(t)/h(t), \dots, x_n = h_n(t)/h(t), f(t) = 0 \quad (h_i, h, f \in \mathbb{R}[t]),$$

which requires the solution of a *single univariate* polynomial:  $f(t) = 0$  (see [37]).

## Symbolic-Numeric Methods

Motivated by the great success of numerical linear algebra, a new trend in applied mathematics is to carefully combine symbolic methods (mostly border bases methods) with numerical calculations, such as singular value decomposition, LU-factorization and other workhorses of numerical linear algebra in order to derive powerful algorithms for large scale problems (see e.g. [30] for details). As mentioned above, symbolic methods are able to transform the given system  $h_i = 0$  into a new, better structured system  $g_j = 0$ . Then the task of computing the complex roots is reduced to (numerical) linear algebra, like computing the eigenvalues/eigenvectors of companion matrices (cf. Sect. 2.2.2 below), or univariate root finding.

## Numerical Methods

The most successful approach in this class of methods is *homotopy continuation*. Such methods rely on Bertini's theorem allowing to deform an easier instance with known solutions of the class of problems to be solved into the original system, without encountering singularities along the path (cf. [39] for details). Keeping track of the roots during this deformation allows to compute the desired roots.

### 2.1.1.2 Existing Methods over the Real Numbers

While the task of solving polynomial equations over the complex numbers is relatively well understood, computing *only the real roots* is still largely open. The need for methods tailored to real root finding is mainly motivated by applications, where often only the real roots are meaningful, and whose number is typically much smaller than the total number of complex solutions. As an illustration, just consider the simple equation  $x_1^2 + x_2^2 = 0$ , where not even the dimensions of the real and complex solution sets agree!

So far, real solving methods were mostly build upon local methods combined with a bisection search strategy. More recently, two new global approaches have been considered which can be seen as refinements of complex root finding methods mentioned above: the SDP based moment approach (which is the focus of this chapter), and a new homotopy continuation method tuned to real roots. The three main classes of methods for real roots are briefly discussed below.

## Subdivision Methods

Combining exclusion criteria to remove parts of the search space not containing any real root and identify regions containing isolated real roots, with local search strategies such as Newton–Raphson or higher order methods, are the basis for the class of subdivision methods. The search space is subdivided until it contains

only a single root and Newton's method converges (cf. e.g. [31] for a recent account). Exclusion criteria include real root counting techniques based e.g. on Sturm-Habicht sequences, Descartes' rule of signs (for univariate polynomials), or signatures of Hermite forms (in the multivariate case). Such techniques, combined with deformation techniques using Puiseux series, are also extended to the problem of computing at least one point in each connected component of an algebraic variety (possibly of positive dimension) (cf. [2] for a detailed account).

### Khovanskii-Rolle Continuation

This method is a recent extension of curve following methods (like homotopy continuation for complex roots) tailored to real roots. It exploits the fact that there are sharp bounds for the number of real roots of systems of equations with few monomials, combined with Gale duality. The approach allows to track significantly fewer paths of an auxiliary system leading to all nondegenerate real solutions of the original system. It is still under investigation, but has the potential to become an efficient algorithm for real root finding (see [3, 40] for details).

### Moment Methods

This class of methods was first proposed in [17] with extensions in [18, 19], and is the focus of this chapter. The basic idea is to compute the real roots by working in a smaller quotient space, obtained by taking the quotient by the *real radical ideal*  $\sqrt{\mathbb{R}I}$  of the original ideal  $I$ , consisting of all polynomials that vanish at the set of common real roots of the original system  $h_i = 0$ . In this way, computing the real roots is again reduced to a task of numerical linear algebra, now in the finite dimensional vector space  $\mathbb{R}[x]/\sqrt{\mathbb{R}I}$  (assuming only that the number of real roots is finite, while the total number of complex roots could be infinite). Finding the real radical ideal is achieved by computing the kernel of a generic moment matrix obtained by solving iteratively certain semidefinite programming problems.

#### 2.1.2 The Basic Idea of the Moment Method

Most symbolic and symbolic/numeric algorithms for solving a system of polynomials decompose the structure of the polynomial ring into its ideal structure (namely, the ideal  $I$  generated by the equations to be solved) and its vector space structure (corresponding to the quotient of the polynomial ring by this ideal). While the former is treated with symbolic methods one can use efficient linear algebra for the latter. We start with an elementary introduction. Let

$$h_1(x) = \cdots = h_m(x) = 0 \tag{2.1}$$

be the system of polynomial equations to be solved. Denote by  $D \in \mathbb{N}$  the maximum degree of the polynomials  $h_i$  and let  $I = \langle h_1, \dots, h_m \rangle$  be the ideal generated by these polynomials, i.e., the set of all polynomials  $\sum_i u_i h_i$  with  $u_i \in \mathbb{R}[x]$ . If we form the matrix  $H$  whose rows are the coefficient vectors of the polynomials  $h_i$ , then the roots of the system (2.1) are precisely the elements  $x \in \mathbb{C}^n$  satisfying  $H[x]_D = 0$ , where for any integer  $t \in \mathbb{N}$ ,

$$[x]_t = (1, x_1, \dots, x_n, x_1^2, x_1 x_2, \dots, x_n^t)$$

denotes the vector of all monomials of degree at most  $t$ . Augmenting the system (2.1) with new polynomials obtained by multiplying the  $h_i$ 's by monomials does not change its set of common roots. Given an integer  $t$ , we add all possible multiples of the  $h_i$ 's with degree at most  $t$ , i.e., we add all ‘valid’ equations:  $x^\alpha h_i = 0$  where  $|\alpha| \leq t - \deg(h_i)$ . This yields a new, larger system of polynomials whose coefficient vectors make the rows of a matrix  $\tilde{H}_t$  (known as Sylvester or Macaulay-like matrix). Again, the roots of (2.1) are those elements  $x \in \mathbb{C}^n$  satisfying  $\tilde{H}_t[x]_t = 0$ .

The basic idea is to *linearize* this system of equations by introducing variables  $y = (y_\alpha)$  for the monomials  $x^\alpha$  and to solve instead a *linear system*:

$$\tilde{H}_t y = 0. \quad (2.2)$$

The kernel of the matrix  $\tilde{H}_t$  is a linear subspace, which contains the vectors  $[x]_t$  for all roots  $x$  of the system (2.1) and thus also their linear span. When the system (2.1) has finitely many complex roots, it turns out that, for  $t$  large enough, (some projection of) the kernel of  $\tilde{H}_t$  coincides with the linear span of the monomial vectors corresponding to the roots of (2.1), which opens the way to extracting the roots. More precisely, the central observation (dating back to [23]) is that for  $t$  large enough a Gaussian elimination on the Sylvester matrix  $\tilde{H}_t$  will reveal a Gröbner basis for the ideal  $I$  and thus the desired quotient ring structure  $\mathbb{R}[x]/I$ . This in turn can be used to reduce the multivariate root finding problem to a simple eigenvalue calculation (as recalled in Sect. 2.2.2).

If we want to compute the real roots only, we need a mechanism to cancel out all (or as many as possible) nonreal solutions among the complex ones. This cancellation can be done by augmenting the original system (2.1) with additional polynomials derived from sums of squares of polynomials in the ideal  $I$ . We introduce this idea by means of a simple example.

*Example 2.1.* Consider the ideal  $I \subseteq \mathbb{R}[x_1, x_2]$  generated by the polynomial  $h = x_1^2 + x_2^2$ . The complex variety is positive dimensional, since it consists of infinitely many complex roots:  $x_2 = \pm ix_1$  ( $x_1 \in \mathbb{C}$ ), while the origin  $(0, 0)$  is the only real root. If we add the two polynomials  $p_1 = x_1, p_2 = x_2$  to  $I$  the real variety remains unchanged, but none of the complex nonreal roots survives this intersection. Note that  $p_1, p_2$  have the property that the polynomial  $p_1^2 + p_2^2 = h$  is a sum of squares of polynomials belonging to  $I$ .

This example illustrates the following fact: If the  $p_i$ 's are polynomials for which  $\sum_i p_i^2 \in I$ , then each  $p_i$  vanishes at all the real roots of the ideal  $I$  (but not necessarily at its complex nonreal roots!). Thus we can add the  $p_i$ 's to the original system (2.1) without altering its set of real roots. The formal tool behind this augmentation is the Real Nullstellensatz (see Theorem 2.1), which states that the set of real solutions to the system (2.1) remains unchanged if we add to it any polynomial appearing with an even degree in a sum of squares polynomial that belongs to  $I$ . The set of all such polynomials is known as the *real radical ideal* of  $I$ , denoted as  $\sqrt{\mathbb{R}I}$  (see Sect. 2.2 for definitions). A main feature of the moment matrix method is that it permits to generate the polynomials in the real radical ideal in a systematic way, using duality.

Let us first look directly at the additional properties that are satisfied by a vector  $y = [x]_t \in \text{Ker } \tilde{H}_t$ , when  $x$  is a *real* root of (2.1). Obviously the matrix  $[x]_s [x]_s^T$  is positive semidefinite for any integer  $s$  and by ‘linearizing’ (replacing  $x^\alpha$  by  $y_\alpha$ ) we obtain the following matrix of generalized Hankel type:  $M_s(y) = (y_{\alpha+\beta})_{\alpha,\beta \in \mathbb{N}_s^n}$ . Matrices with this generalized Hankel structure are also known as *moment matrices* (see Definition 2.3). As an illustration we display  $M_s(y)$  for the case  $n = 2$ :

$$[x]_s [x]_s^T = \begin{pmatrix} 1 & x_1 & x_2 & x_1^2 & \dots \\ x_1 & x_1^2 & x_1 x_2 & x_1^3 & \dots \\ x_2 & x_1 x_2 & x_2^2 & x_1^2 x_2 & \dots \\ x_1^2 & x_1^3 & x_1^2 x_2 & x_1^4 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \rightsquigarrow M_s(y) = \begin{pmatrix} 1 & y_{10} & y_{01} & y_{20} & \dots \\ y_{10} & y_{20} & y_{11} & y_{30} & \dots \\ y_{01} & y_{11} & y_{02} & y_{21} & \dots \\ y_{20} & y_{30} & y_{21} & y_{40} & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

Therefore, we can restrict the search in the kernel of the Sylvester matrix  $\tilde{H}_t$  to the vectors  $y$  satisfying the additional positive semidefiniteness condition:  $M_s(y) \geq 0$  for all  $s \leq t/2$ . This condition captures precisely the ‘real algebraic’ nature of real numbers vs. complex numbers, as it would not be valid for vectors  $y$  corresponding to complex nonreal roots.

*Example 2.2 (Example 2.1 cont).* Say we wish to compute the real roots of the polynomial  $h = x_1^2 + x_2^2$ . After linearization, the constraint  $Hy = 0$  reads:  $y_{20} + y_{02} = 0$ . Positive semidefiniteness requires  $y_{20} \geq 0$ ,  $y_{02} \geq 0$  which, combined with  $y_{20} + y_{02} = 0$  implies  $y_{20} = y_{02} = 0$  and thus  $y_{10} = y_{01} = y_{11} = 0$  (using again  $M_1(y) \geq 0$ ). Therefore, we find  $y = (1, 0, 0, 0, 0, 0)$  as the unique solution, so that  $y = [x]_2$  corresponds to the unique real root  $x = (0, 0)$  of  $h$ . The kernel of  $M_1(y)$  contains the vectors  $(0, 1, 0)$  and  $(0, 0, 1)$ , which can be seen as the coefficient vectors of the two polynomials  $p_1 = x_1$  and  $p_2 = x_2$  in the monomial basis  $\{1, x_1, x_2\}$  of  $\mathbb{R}[x]_1$ . In other words the kernel of  $M_1(y)$  already contains a basis of the real radical ideal  $\sqrt{\mathbb{R}I}$ .

Although the above example is extremely simplistic, it conveys the main idea: The kernel of  $M_s(y)$  characterizes (for  $s$  large enough) the real radical ideal and plays the role of the range space of  $H$  in standard normal form algorithms.

### 2.1.3 Organization of the Chapter

First we recall some basic material from polynomial algebra in Sect. 2.2. This material can be found in most standard textbooks and is used throughout the chapter. The relation between moment matrices and real radical ideals as well as the moment method for real root finding is discussed in Sect. 2.3. This section and in particular the semidefinite characterization of the real radical ideal form the heart of the chapter. We also discuss the link to some complex root finding methods and in Sect. 2.4 we briefly touch some related topics: polynomial optimization and the study of semi-algebraic sets, emptiness certificates, positive dimensional varieties, and quotient ideals. Throughout the chapter we illustrate the results with various examples.

## 2.2 Preliminaries of Polynomial Algebra

### 2.2.1 Polynomial Ideals and Varieties

#### 2.2.1.1 The Polynomial Ring and Its Dual

For the sake of simplicity we deal with polynomials with real coefficients only although some results remain valid for polynomials with complex coefficients. Throughout  $\mathbb{R}[x] := \mathbb{R}[x_1, \dots, x_n]$  denotes the ring of multivariate polynomials in  $n$  variables. For  $\alpha \in \mathbb{N}^n$ ,  $x^\alpha$  denotes the monomial  $x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ , with degree  $|\alpha| := \sum_{i=1}^n \alpha_i$ . Set  $\mathbb{N}_t^n := \{\alpha \in \mathbb{N}^n \mid |\alpha| \leq t\}$  and let

$$[x]_\infty = (x^\alpha)_{\alpha \in \mathbb{N}^n}, \quad [x]_t = (x^\alpha)_{\alpha \in \mathbb{N}_t^n}$$

denote the vectors comprising all monomials (resp., all monomials of degree at most  $t$ ) in  $n$  variables. A polynomial  $p \in \mathbb{R}[x]$  can be written as  $p = \sum_{\alpha \in \mathbb{N}^n} p_\alpha x^\alpha$  with finitely many nonzero  $p_\alpha$ 's; its support is the set of monomials appearing with a nonzero coefficient, its (total) degree  $\deg(p)$  is the largest degree of a monomial in the support of  $p$ , and  $\text{vec}(p) = (p_\alpha)$  denotes the vector of coefficients of  $p$ . The set  $\mathbb{R}[x]_t$  consists of all polynomials with degree at most  $t$ .

Given a vector space  $A$  on  $\mathbb{R}$ , its dual space  $A^*$  consists of all linear functionals from  $A$  to  $\mathbb{R}$ . The orthogonal complement of a subset  $B \subseteq A$  is

$$B^\perp := \{L \in A^* \mid L(b) = 0 \ \forall b \in B\}$$

and  $\text{Span}_{\mathbb{R}}(B)$  denotes the linear span of  $B$ . Then,  $\text{Span}_{\mathbb{R}}(B) \subseteq (B^\perp)^\perp$ , with equality when  $A$  is finite dimensional. We consider here the cases  $A = \mathbb{R}[x]$  and  $A = \mathbb{R}[x]_t$ . Examples of linear functionals on  $\mathbb{R}[x]$  are the *evaluation*

$$\Lambda_v : p \in \mathbb{R}[x] \mapsto \Lambda_v(p) = p(v) \tag{2.3}$$

at a point  $v \in \mathbb{R}^n$  and, more generally, the differential functional

$$\partial_v^\alpha : p \in \mathbb{R}[x] \mapsto \partial_v^\alpha(p) = \frac{1}{\prod_{i=1}^n \alpha_i!} \left( \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \dots \partial x_n^{\alpha_n}} p \right)(v), \quad (2.4)$$

which evaluates at  $v \in \mathbb{R}^n$  the (scaled)  $\alpha$ -th derivative of  $p$  (where  $\alpha \in \mathbb{N}$ ). For  $\alpha = 0$ ,  $\partial_v^\alpha$  coincides with the evaluation at  $v$ , i.e.,  $\partial_v^0 = \Lambda_v$ . For  $\alpha, \beta \in \mathbb{N}^n$ ,

$$\partial_0^\alpha(x^\beta) = 1 \text{ if } \alpha = \beta, \text{ and } 0 \text{ otherwise.}$$

Therefore, any linear form  $\Lambda \in \mathbb{R}[x]^*$  can be written in the form:

$$\Lambda = \sum_{\alpha \in \mathbb{N}^n} \Lambda(x^\alpha) \partial_0^\alpha.$$

This is in fact a formal power series as in general infinitely many  $\Lambda(x^\alpha)$  are nonzero. Let  $y = (y_\alpha)$  denote the coefficient series of  $\Lambda$  in  $(\partial_0^\alpha)$ , i.e.,  $y_\alpha = \Lambda(x^\alpha)$ , such that  $\Lambda(p) = y^T \text{vec}(p)$  for all  $p \in \mathbb{R}[x]$ . For instance, the evaluation at  $v \in \mathbb{R}^n$  reads  $\Lambda_v = \sum_\alpha v^\alpha \partial_0^\alpha$ , with coefficient series  $[v]_\infty = (v^\alpha)_{\alpha \in \mathbb{N}^n}$  in  $(\partial_0^\alpha)$ .

### 2.2.1.2 Ideals and Varieties

A linear subspace  $I \subseteq \mathbb{R}[x]$  is an *ideal* if  $p \in I$ ,  $q \in \mathbb{R}[x]$  implies  $pq \in I$ . The *ideal generated* by  $h_1, \dots, h_m \in \mathbb{R}[x]$  is defined as

$$I = \langle h_1, \dots, h_m \rangle := \left\{ \sum_{j=1}^m u_j h_j \mid u_1, \dots, u_m \in \mathbb{R}[x] \right\}$$

and the set  $\{h_1, \dots, h_m\}$  is then called a *basis* of  $I$ . By the finite basis theorem [6, Sect. 2.5, Theorem 4], every ideal in  $\mathbb{R}[x]$  admits a finite basis. Given an ideal  $I \subseteq \mathbb{R}[x]$ , the *algebraic variety* of  $I$  is the set

$$V_{\mathbb{C}}(I) = \{v \in \mathbb{C}^n \mid h_j(v) = 0 \ \forall j = 1, \dots, m\}$$

of common complex zeros to all polynomials in  $I$  and its *real variety* is

$$V_{\mathbb{R}}(I) := V_{\mathbb{C}}(I) \cap \mathbb{R}^n.$$

The ideal  $I$  is said to be *zero-dimensional* when its complex variety  $V_{\mathbb{C}}(I)$  is finite. The *vanishing ideal* of a subset  $V \subseteq \mathbb{C}^n$  is the ideal

$$\mathcal{I}(V) := \{f \in \mathbb{R}[x] \mid f(v) = 0 \ \forall v \in V\}.$$

For an ideal  $I \subseteq \mathbb{R}[x]$ , we may also define the ideal

$$\sqrt{I} := \left\{ f \in \mathbb{R}[x] \mid f^m \in I \text{ for some } m \in \mathbb{N} \setminus \{0\} \right\},$$

called the *radical ideal* of  $I$ , and the *real radical ideal* (or *real ideal*)

$$\sqrt{\mathbb{R}I} := \left\{ p \in \mathbb{R}[x] \mid p^{2m} + \sum_j q_j^2 \in I \text{ for some } q_j \in \mathbb{R}[x], m \in \mathbb{N} \setminus \{0\} \right\}.$$

An ideal  $I$  is said to be *radical* (resp., *real radical*) if  $I = \sqrt{I}$  (resp.,  $I = \sqrt{\mathbb{R}I}$ ). For instance, the ideal  $I = \langle x_1^2 + x_2^2 \rangle$  is not real radical since  $x_1, x_2 \in \sqrt{\mathbb{R}I} \setminus I$ . As can be easily verified,  $I$  is radical if and only if  $p^2 \in I$  implies  $p \in I$ , and  $I$  is real radical if and only if  $\sum_i p_i^2 \in I$  implies  $p_i \in I$  for all  $i$ . We have the following chains of inclusion:

$$I \subseteq \sqrt{I} \subseteq \mathcal{I}(V_{\mathbb{C}}(I)), \quad I \subseteq \sqrt{\mathbb{R}I} \subseteq \mathcal{I}(V_{\mathbb{R}}(I)).$$

The relation between vanishing and (real) radical ideals is stated in the following two famous theorems:

**Theorem 2.1.** *Let  $I \subseteq \mathbb{R}[x]$  be an ideal.*

- (i) **Hilbert's Nullstellensatz** (see, e.g., [6, Sect. 4.1]) *The radical ideal of  $I$  is equal to the vanishing ideal of its variety, i.e.,  $\sqrt{I} = \mathcal{I}(V_{\mathbb{C}}(I))$ .*
- (ii) **Real Nullstellensatz** (see, e.g., [4, Sect. 4.1]) *The real radical ideal of  $I$  is equal to the vanishing ideal of its real variety, i.e.,  $\sqrt{\mathbb{R}I} = \mathcal{I}(V_{\mathbb{R}}(I))$ .*

## 2.2.2 The Eigenvalue Method for Complex Roots

### 2.2.2.1 The Quotient Space $\mathbb{R}[x]/I$

The quotient set  $\mathbb{R}[x]/I$  consists of all cosets  $[f] := f + I = \{f + q \mid q \in I\}$  for  $f \in \mathbb{R}[x]$ , i.e. all equivalent classes of polynomials in  $\mathbb{R}[x]$  modulo  $I$ . This quotient set  $\mathbb{R}[x]/I$  is an algebra with addition  $[f] + [g] := [f + g]$ , scalar multiplication  $\lambda[f] := [\lambda f]$  and multiplication  $[f][g] := [fg]$ , for  $\lambda \in \mathbb{R}$ ,  $f, g \in \mathbb{R}[x]$ . The following classical result relates the dimension of  $\mathbb{R}[x]/I$  and the cardinality of the variety  $V_{\mathbb{C}}(I)$  (see e.g. [6, 42]).

**Theorem 2.2.** *Let  $I$  be an ideal in  $\mathbb{R}[x]$ . Then,*

$$|V_{\mathbb{C}}(I)| < \infty \iff \dim \mathbb{R}[x]/I < \infty.$$

Moreover,  $|V_{\mathbb{C}}(I)| \leq \dim \mathbb{R}[x]/I$ , with equality if and only if  $I$  is radical.

Assume that the number of complex roots is finite and set  $N := \dim \mathbb{R}[x]/I$ , so that  $|V_{\mathbb{C}}(I)| \leq N < \infty$ . Consider a set  $\mathcal{B} := \{b_1, \dots, b_N\} \subseteq \mathbb{R}[x]$  for which the cosets  $[b_1], \dots, [b_N]$  are pairwise distinct and  $\{[b_1], \dots, [b_N]\}$  is a (linear) basis of  $\mathbb{R}[x]/I$ . By abuse of language we also say that  $\mathcal{B}$  itself is a basis of  $\mathbb{R}[x]/I$ . Then every  $f \in \mathbb{R}[x]$  can be written in a unique way as  $f = \sum_{i=1}^N c_i b_i + p$ , where  $c_i \in \mathbb{R}$  and  $p \in I$ . The polynomial

$$\mathcal{N}_{\mathcal{B}}(f) := \sum_{i=1}^N c_i b_i$$

is called the *normal form* of  $f$  modulo  $I$  with respect to the basis  $\mathcal{B}$ . In other words, we have the direct sum decomposition:

$$\mathbb{R}[x] = \text{Span}_{\mathbb{R}}(\mathcal{B}) \oplus I,$$

and  $\text{Span}_{\mathbb{R}}(\mathcal{B})$  and  $\mathbb{R}[x]/I$  are isomorphic vector spaces. We now introduce the eigenvalue method for computing all roots of a zero-dimensional ideal, which we first describe in the univariate case.

### 2.2.2.2 Computing Roots with Companion Matrices

Consider first a univariate polynomial  $p = x^d - a_{d-1}x^{d-1} - \dots - a_1x - a_0$  and the ideal  $I = \langle p \rangle$ . Then the set  $\mathcal{B} = \{1, x, \dots, x^{d-1}\}$  is a basis of  $\mathbb{R}[x]/I$ . The following matrix

$$X := \begin{pmatrix} 0 & a_0 \\ I_{d-1} & a \end{pmatrix} \quad \text{where } a = (a_1, \dots, a_{d-1})^T,$$

is known as the *companion matrix* of the polynomial  $p$ . One can easily verify that  $\det(X - xI) = (-1)^d p(x)$ , so that the eigenvalues of  $X$  are precisely the roots of the polynomials  $p$ . Therefore the roots of a univariate polynomial can be found with an eigenvalue computation. Moreover, the columns of the companion matrix  $X$  correspond to the normal forms of the monomials in  $x\mathcal{B} = \{x, x^2, \dots, x^d\}$  modulo  $I$  with respect to the basis  $\mathcal{B}$ . As we now see these facts extend naturally to the multivariate case.

Given  $h \in \mathbb{R}[x]$ , we define the *multiplication (by  $h$ ) operator* in  $\mathbb{R}[x]/I$  as

$$\begin{aligned} X_h : \mathbb{R}[x]/I &\longrightarrow \mathbb{R}[x]/I \\ [f] &\longmapsto X_h([f]) := [hf], \end{aligned} \tag{2.5}$$

which can be represented by its matrix (again denoted  $X_h$  for simplicity) with respect to the basis  $\mathcal{B}$  of  $\mathbb{R}[x]/I$ . Namely, if we set  $\mathcal{N}_{\mathcal{B}}(hb_j) := \sum_{i=1}^N a_{ij}b_i$  (where  $a_{ij} \in \mathbb{R}$ ), then the  $j$ th column of  $X_h$  is the vector  $(a_{ij})_{i=1}^N$ . Note also that, since  $hb_j - \mathcal{N}_{\mathcal{B}}(hb_j) \in I$ , polynomials in  $I$  can be read directly from  $X_h$ . This fact will

play an important role for border bases (see Sect. 2.2.3). In the univariate case, when  $I = \langle p \rangle$  and  $h = x$ , the multiplication matrix  $X_x$  is precisely the companion matrix  $X$  of  $p$  introduced above. Throughout we also denote by  $X_i := X_{x_i}$  the multiplication operator by the variable  $x_i$  in the multivariate case.

The following famous result (see e.g. [5, Chap. 2, Sect. 4]) relates the eigenvalues of the multiplication operators in  $\mathbb{R}[x]/I$  to the algebraic variety  $V_{\mathbb{C}}(I)$ . This result underlies the well known *eigenvalue method*, which plays a central role in many algorithms for complex root solving.

**Theorem 2.3 (Stickelberger theorem).** *Let  $I$  be a zero-dimensional ideal in  $\mathbb{R}[x]$ , let  $\mathcal{B}$  be a basis of  $\mathbb{R}[x]/I$ , and let  $h \in \mathbb{R}[x]$ . The eigenvalues of the multiplication operator  $X_h$  are the evaluations  $h(v)$  of the polynomial  $h$  at the points  $v \in V_{\mathbb{C}}(I)$ . Moreover, for all  $v \in V_{\mathbb{C}}(I)$ ,*

$$(X_h)^T[v]_{\mathcal{B}} = h(v)[v]_{\mathcal{B}},$$

setting  $[v]_{\mathcal{B}} = (b(v))_{b \in \mathcal{B}}$ ; that is, the vector  $[v]_{\mathcal{B}}$  is a left eigenvector of the multiplication operator with eigenvalue  $h(v)$ .

Therefore the eigenvalues of the matrices  $X_i$  are the  $i$ th coordinates of the points  $v \in V_{\mathbb{C}}(I)$ , which can be derived from the left eigenvectors  $[v]_{\mathcal{B}}$ . Practically, one can recover the roots from the left eigenvectors when the eigenspaces of  $X_h^T$  all have dimension one. This is the case when the values  $h(v)$  ( $v \in V_{\mathbb{C}}(I)$ ) are pairwise distinct (easy to achieve, e.g., if we choose  $h$  to be a generic linear form) and when the ideal  $I$  is radical (since the dimension of  $\mathbb{R}[x]/I$  is then equal to the number of roots so that the vectors  $[v]_{\mathcal{B}}$  ( $v \in V_{\mathbb{C}}(I)$ ) form a complete basis of eigenvectors).

Summarizing, the task of solving a system of polynomial equations is reduced to a task of numerical linear algebra once a basis of  $\mathbb{R}[x]/I$  and a normal form algorithm are available, as they permit the construction of the multiplication matrices  $X_i, X_h$ . Moreover, the roots  $v \in V_{\mathbb{C}}(I)$  can be successfully constructed from the eigenvectors/eigenvalues of  $X_h$  when  $I$  is radical and  $h$  is generic. Our strategy for computing the real variety  $V_{\mathbb{R}}(I)$  will be to compute a linear basis of the quotient space  $\mathbb{R}[x]/\sqrt{I}$  and the corresponding multiplication matrices, so that we can apply the eigenvalue method precisely in this setting of having a radical (even real radical) ideal.

The number of (real) roots can be counted using Hermite's quadratic form:

$$\begin{aligned} S_h : \mathbb{R}[x]/I \times \mathbb{R}[x]/I &\rightarrow \mathbb{R} \\ ([f], [g]) &\mapsto \text{Tr}(X_{fgh}). \end{aligned}$$

Here,  $\text{Tr}(X_{fgh})$  is the trace of the multiplication (by the polynomial  $fgh$ ) matrix. As  $S_h$  is a symmetric matrix, all its eigenvalues are real. Denote by  $\sigma_+(S_h)$  (resp.,  $\sigma_-(S_h)$ ) its number of positive (resp., negative) eigenvalues. The following classical result shows how to count the number of roots satisfying prescribed sign conditions (cf. e.g. [2]).

**Theorem 2.4.** Let  $I \subseteq \mathbb{R}[x]$  be a zero-dimensional ideal and  $h \in \mathbb{R}[x]$ . Then,

$$\text{rank } S_h = |\{v \in V_{\mathbb{C}}(I) \mid h(v) \neq 0\}|,$$

$$\sigma_+(S_h) - \sigma_-(S_h) = |\{v \in V_{\mathbb{R}}(I) \mid h(v) > 0\}| - |\{v \in V_{\mathbb{R}}(I) \mid h(v) < 0\}|.$$

In particular, for the constant polynomial  $h = 1$ ,

$$\text{rank}(S_1) = |V_{\mathbb{C}}(I)| \text{ and } \sigma_+(S_1) - \sigma_-(S_1) = |V_{\mathbb{R}}(I)|.$$

### 2.2.3 Border Bases and Normal Forms

The eigenvalue method for solving polynomial equations (described in the preceding section) requires the knowledge of a basis of  $\mathbb{R}[x]/I$  and of an algorithm to compute the normal form of a polynomial with respect to this basis.

A well known basis of  $\mathbb{R}[x]/I$  is the set of standard monomials with respect to some monomial ordering. The classical way to find standard monomials is to construct a Gröbner basis of  $I$  (then the standard monomials are the monomials not divisible by any leading monomial of a polynomial in the Gröbner basis). Moreover, once a Gröbner basis is known, the normal form of a polynomial can be found via a polynomial division algorithm (see, e.g., [6, Chap. 1] for details). Other techniques have been proposed, producing more general bases which do not depend on a specific monomial ordering and often are numerically more stable. In particular, algorithms have been proposed for constructing border bases of  $I$  leading to general (connected to 1) bases of  $\mathbb{R}[x]/I$  (see [9, Chap. 4], [14, 29, 42]); these objects are introduced below. The moment matrix approach for computing real roots presented in this chapter leads naturally to the computation of such general bases.

**Definition 2.1.** Given a set  $\mathcal{B}$  of monomials, define the new sets of monomials

$$\mathcal{B}^+ := \mathcal{B} \cup \bigcup_{i=1}^n x_i \mathcal{B} = \mathcal{B} \cup \{x_i b \mid b \in \mathcal{B}, i = 1, \dots, n\}, \quad \partial \mathcal{B} = \mathcal{B}^+ \setminus \mathcal{B},$$

called, respectively, the *one-degree prolongation* of  $\mathcal{B}$  and the *border* of  $\mathcal{B}$ . The set  $\mathcal{B}$  is said to be *connected to 1* if  $1 \in \mathcal{B}$  and each  $m \in \mathcal{B} \setminus \{1\}$  can be written as  $m = x_{i_1} \dots x_{i_k}$  with  $x_{i_1}, x_{i_1} x_{i_2}, \dots, x_{i_1} \dots x_{i_k} \in \mathcal{B}$ . Moreover,  $\mathcal{B}$  is said to be *stable by division* if all divisors of  $m \in \mathcal{B}$  also belong to  $\mathcal{B}$ . Obviously,  $\mathcal{B}$  is connected to 1 if it is stable by division.

Assume  $\mathcal{B}$  is a set of monomials which is connected to 1. For each border monomial  $m \in \partial \mathcal{B}$ , consider a polynomial  $f_m$  of the form

$$f_m := m - r_m, \quad \text{where } r_m \in \text{Span}_{\mathbb{R}}(\mathcal{B}). \tag{2.6}$$

The family  $F := \{f_m \mid m \in \partial\mathcal{B}\}$  is called a *rewriting family for  $\mathcal{B}$*  in [30, 32]. Using  $F$ , one can express all border monomials in  $\partial\mathcal{B}$  as linear combinations of monomials in  $\mathcal{B}$  modulo the ideal  $\langle F \rangle$ . Moreover, the rewriting family  $F$  can be used in a division algorithm to rewrite any polynomial  $p \in \mathbb{R}[x]$  as

$$p = r + \sum_{m \in \partial\mathcal{B}} u_m f_m, \quad \text{where } r \in \text{Span}_{\mathbb{R}}(\mathcal{B}), \quad u_m \in \mathbb{R}[x]. \quad (2.7)$$

This expression is in general not unique, as it depends on the order in which the polynomials of  $F$  are used throughout the division process.

*Example 2.3.* Let  $\mathcal{B} = \{1, x_1, x_2\}$  with border set  $\partial\mathcal{B} = \{x_1^2, x_1 x_2, x_2^2\}$ , and consider the rewriting family

$$F = \left\{ f_{x_1^2} = x_1^2 + 1, f_{x_1 x_2} = x_1 x_2 - 1, f_{x_2^2} = x_2^2 + 1 \right\}.$$

There are two possibilities to rewrite the polynomial  $p = x_1^2 x_2$ . Either, first divide by  $f_{x_1 x_2}$  and obtain  $p = x_1^2 x_2 = x_1 f_{x_1 x_2} + x_1$  with  $r = x_1$ , or first divide by  $f_{x_1^2}$  and obtain  $p = x_1^2 x_2 = x_2 f_{x_1^2} - x_2$  with  $r = -x_2$ .

In view of (2.7), the set  $\mathcal{B}$  spans the vector space  $\mathbb{R}[x]/\langle F \rangle$ , but is in general not linearly independent. Linear independence guarantees uniqueness of the decomposition (2.7) and, as Theorem 2.5 below shows, is equivalent to the commutativity of certain formal multiplication operators.

Consider the linear operator  $X_i : \text{Span}_{\mathbb{R}}(\mathcal{B}) \rightarrow \text{Span}_{\mathbb{R}}(\mathcal{B})$  defined using the rewriting family  $F$ , namely, for  $b \in \mathcal{B}$ ,

$$X_i(b) = \begin{cases} x_i b & \text{if } x_i b \in \mathcal{B}, \\ x_i b - f_{x_i b} = r_{x_i b} & \text{otherwise,} \end{cases}$$

and extend  $X_i$  to  $\text{Span}_{\mathbb{R}}(\mathcal{B})$  by linearity. Denote also by  $X_i$  the matrix of this linear operator, which can be seen as a *formal multiplication (by  $x_i$ ) matrix*.

**Theorem 2.5 ([29]).** *Let  $F$  be a rewriting family for a set  $\mathcal{B}$  of monomials connected to 1, and consider the ideal  $J := \langle F \rangle$ . The following conditions are equivalent:*

- (i) *The formal multiplication matrices  $X_1, \dots, X_n$  commute pairwise.*
- (ii) *The set  $\mathcal{B}$  is a (linear) basis of  $\mathbb{R}[x]/J$ , i.e.,  $\mathbb{R}[x] = \text{Span}_{\mathbb{R}}(\mathcal{B}) \oplus J$ .*

*Then, the set  $F$  is said to be a border basis of the ideal  $J$ , and the matrix  $X_i$  represents the multiplication operator by  $x_i$  in  $\mathbb{R}[x]/J$  with respect to  $\mathcal{B}$ .*

This theorem is the crucial tool for efficient root finding algorithms based on normal form reductions, which iteratively construct a system of polynomial equations giving a rewriting family corresponding to a commuting family of multiplication matrices (thus reducing the root finding problem to an eigenvalue computation, see [30]). We illustrate Theorem 2.5 on a small example.

*Example 2.4.* Let  $\mathcal{B} = \{1, x_1\}$  with border set  $\partial\mathcal{B} = \{x_2, x_1x_2, x_1^2\}$ , and consider the rewriting family

$$F = \left\{ f_{x_1^2} = x_1^2 + 1, f_{x_1x_2} = x_1x_2 - 1, f_{x_2} = x_2 + x_1 \right\}.$$

As  $x_1 \in \mathcal{B}$ ,  $x_1^2 = f_{x_1^2} - 1$ ,  $x_2 = f_{x_2} - x_1$ , and  $x_2x_1 = f_{x_1x_2} + 1$ , we have

$$X_1 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \quad X_2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

As the formal multiplication matrices  $X_1, X_2$  commute, we can conclude that  $F$  is a border basis of  $\langle F \rangle$  and  $\mathbb{R}[x] = \text{Span}_{\mathbb{R}}(\mathcal{B}) \oplus \langle F \rangle$ .

## 2.3 The Moment Method for Real Root Finding

We just saw that computing the complex roots of an ideal can be reduced to an eigenvalue computation. This technique applies only when the number of complex roots is finite, and involves matrices whose size is at least the number of complex roots. However, in most applications one is only interested in the real roots, whose number can be a very small fraction of the total number of roots. Therefore one needs a tool to isolate the real roots from the complex nonreal ones. As we briefly mentioned in the Introduction, a possible strategy is to add new polynomials from the real radical ideal to the original system to be solved. To find these polynomials in a systematic way we propose to work on the ‘dual side’, i.e., to consider linear forms  $\Lambda$  on the polynomial ring  $\mathbb{R}[x]$  or its subspaces  $\mathbb{R}[x]_t$  of bounded degree. Indeed, it turns out that the kernel of such linear forms carries all information about the real radical ideal and the real variety when the linear form is assumed to satisfy some positivity condition. In this section we explain the method in detail and illustrate it on a few examples.

### 2.3.1 Positive Linear Forms and Real Radical Ideals

Given a linear form  $\Lambda \in \mathbb{R}[x]^*$ , consider the quadratic form on  $\mathbb{R}[x]$

$$Q_\Lambda : f \in \mathbb{R}[x] \mapsto Q_\Lambda(f) = \Lambda(f^2) \in \mathbb{R},$$

with kernel  $\text{Ker } Q_\Lambda := \{f \in \mathbb{R}[x] \mid \Lambda(fg) = 0 \ \forall g \in \mathbb{R}[x]\}$ .

**Definition 2.2 (Positivity).**  $\Lambda \in \mathbb{R}[x]^*$  is said to be *positive* if  $\Lambda(f^2) \geq 0$  for all  $f \in \mathbb{R}[x]$ , i.e., if the quadratic form  $Q_\Lambda$  is positive semidefinite.

The following simple lemma provides the link to real radical polynomial ideals.

**Lemma 2.1 ([20,26]).** *Let  $\Lambda \in \mathbb{R}[x]^*$ . Then  $\text{Ker } Q_\Lambda$  is an ideal in  $\mathbb{R}[x]$ , which is real radical when  $\Lambda$  is positive.*

*Proof.*  $\text{Ker } Q_\Lambda$  is obviously an ideal, from its definition. Assume  $\Lambda$  is positive. First we show that, for  $p \in \mathbb{R}[x]$ ,  $\Lambda(p^2) = 0$  implies  $\Lambda(p) = 0$ . Indeed, if  $\Lambda(p^2) = 0$  then, for any scalar  $t \in \mathbb{R}$ , we have:

$$0 \leq \Lambda((p+t)^2) = \Lambda(p^2) + 2t\Lambda(p) + t^2\Lambda(1) = t(2\Lambda(p) + t\Lambda(1)),$$

which implies  $\Lambda(p) = 0$ . Assume now  $\sum_i p_i^2 \in \text{Ker } Q_\Lambda$  for some  $p_i \in \mathbb{R}[x]$ ; we show  $p_i \in \text{Ker } Q_\Lambda$ . For any  $g \in \mathbb{R}[x]$ , we have  $0 = \Lambda(g^2(\sum_i p_i^2)) = \sum_i \Lambda(p_i^2 g^2)$  which, as  $\Lambda(p_i^2 g^2) \geq 0$ , implies  $\Lambda(p_i^2 g^2) = 0$ . By the above, this in turn implies  $\Lambda(p_i g) = 0$ , thus showing  $p_i \in \text{Ker } Q_\Lambda$ . Therefore,  $\text{Ker } Q_\Lambda$  is real radical.  $\square$

We now introduce moment matrices, which permit to reformulate positivity of  $\Lambda$  in terms of positive semidefiniteness of an associated matrix  $M(\Lambda)$ .

**Definition 2.3 (Moment matrix).** A symmetric matrix  $M = (M_{\alpha\beta})$  indexed by  $\mathbb{N}^n$  is said to be a *moment matrix* (or a *generalized Hankel matrix*) if its  $(\alpha, \beta)$ -entry depends only on the sum  $\alpha + \beta$  of the indices. Given  $\Lambda \in \mathbb{R}[x]^*$ , the matrix

$$M(\Lambda) := (\Lambda(x^\alpha x^\beta))_{\alpha, \beta \in \mathbb{N}^n}$$

is called the *moment matrix* of  $\Lambda$ .

If  $y \in \mathbb{R}^{\mathbb{N}^n}$  is the coefficient series of  $\Lambda \in \mathbb{R}[x]^*$ , i.e.,  $\Lambda = \sum_\alpha y_\alpha \partial_0^\alpha$ , then its moment matrix  $M(y) = (y_{\alpha+\beta})_{\alpha, \beta \in \mathbb{N}^n}$  coincides with the moment matrix  $M(\Lambda)$  of  $\Lambda$ . These two definitions are obviously equivalent and, depending on the context, it is more convenient to use  $M(y)$  or  $M(\Lambda)$ .

Note that  $Q_\Lambda(p) = \Lambda(p^2) = \text{vec}(p)^T M(\Lambda) \text{vec}(p)$  for all  $p \in \mathbb{R}[x]$ . Hence,  $M(\Lambda)$  is the matrix of the quadratic form  $Q_\Lambda$  in the monomial base, and  $\Lambda$  is positive if and only if  $M(\Lambda) \succeq 0$ .

Moreover, a polynomial  $p$  belongs to the kernel of  $Q_\Lambda$  if and only if its coefficient vector belongs to  $\text{Ker } M(\Lambda)$ . Throughout we identify polynomials  $p = \sum_\alpha p_\alpha x^\alpha$  with their coefficient vectors  $\text{vec}(p) = (p_\alpha)_\alpha$  and thus  $\text{Ker } Q_\Lambda$  with  $\text{Ker } M(\Lambda)$ . Hence we view  $\text{Ker } M(\Lambda)$  as a set of polynomials. By Lemma 2.1,  $\text{Ker } M(\Lambda)$  is an ideal of  $\mathbb{R}[x]$ , which is real radical when  $M(\Lambda) \succeq 0$ . Moreover, the next lemma shows that  $\text{Ker } M(\Lambda)$  is a zero-dimensional ideal precisely when the matrix  $M(\Lambda)$  has finite rank.

*Example 2.5.* For  $n = 2$ , consider the linear form  $\Lambda \in \mathbb{R}[x]^*$  defined by  $\Lambda(1) = \Lambda(x_1^2) = 1$  and  $\Lambda(x_1^{\alpha_1} x_2^{\alpha_2}) = 0$  for all other monomials. Then  $\Lambda$  is positive,  $\text{rank } M(\Lambda) = 2$  and the kernel of  $M(\Lambda)$  is the ideal  $\langle x_2, 1 - x_1^2 \rangle$ .

**Lemma 2.2.** *Let  $\Lambda \in \mathbb{R}[x]^*$  and let  $\mathcal{B}$  be a set of monomials. Then,  $\mathcal{B}$  indexes a maximal linearly independent set of columns of  $M(\Lambda)$  if and only if  $\mathcal{B}$  corresponds to a basis of  $\mathbb{R}[x]/\text{Ker } M(\Lambda)$ . That is,*

$$\text{rank } M(\Lambda) = \dim \mathbb{R}[x]/\text{Ker } M(\Lambda).$$

Next we collect some properties of the moment matrix of evaluations at points of  $\mathbb{R}^n$ .

**Lemma 2.3.** *If  $\Lambda = \Lambda_v$  is the evaluation at  $v \in \mathbb{R}^n$ , then  $M(\Lambda_v) = [v]_\infty[v]_\infty^T$  has rank 1 and its kernel is  $\mathcal{I}(v)$ , the vanishing ideal of  $v$ . More generally, if  $\Lambda$  is a conic combination of evaluations at real points, say  $\Lambda = \sum_{i=1}^r \lambda_i \Lambda_{v_i}$  where  $\lambda_i > 0$  and  $v_i \in \mathbb{R}^n$  are pairwise distinct, then  $M(\Lambda) = \sum_{i=1}^r \lambda_i [v_i]_\infty[v_i]_\infty^T$  has rank  $r$  and its kernel is  $\mathcal{I}(v_1, \dots, v_r)$ , the vanishing ideal of the  $v_i$ 's.*

The following theorem of Curto and Fialkow [7] shows that any positive linear form  $\Lambda$  with a finite rank moment matrix is a conic combination of evaluations at real points. In other words, it shows that the implication of Lemma 2.3 holds as an equivalence. This result will play a crucial role in our approach. We give a proof, based on [20], although some details are simplified.

**Theorem 2.6 (Finite rank moment matrix theorem [7]).** *Assume that  $\Lambda \in \mathbb{R}[x]^*$  is positive with  $\text{rank } M(\Lambda) =: r < \infty$ . Then,  $\Lambda = \sum_{i=1}^r \lambda_i \Lambda_{v_i}$  for some distinct  $v_1, \dots, v_r \in \mathbb{R}^n$  and some scalars  $\lambda_i > 0$ . Moreover,  $\{v_1, \dots, v_r\} = V_{\mathbb{C}}(\text{Ker } M(\Lambda))$ .*

*Proof.* By Lemma 2.1,  $J := \text{Ker } M(\Lambda)$  is a real radical ideal and, by Lemma 2.2 (combined with Theorem 2.2), the ideal  $J$  is zero-dimensional and satisfies  $\dim \mathbb{R}[x]/J = r$ . Therefore,  $|V_{\mathbb{C}}(J)| = r$  and  $V_{\mathbb{C}}(J) \subseteq \mathbb{R}^n$ . Say,

$$V_{\mathbb{C}}(J) = \{v_1, \dots, v_r\} \subseteq \mathbb{R}^n$$

so that  $J = \mathcal{I}(v_1, \dots, v_r)$  is the vanishing ideal of the  $v_i$ 's. Let  $p_1, \dots, p_r$  be interpolation polynomials at  $v_1, \dots, v_r$ , respectively, that is,  $p_i(v_j) = 1$  if  $i = j$  and 0 otherwise. We first claim:

The set  $\{p_1, \dots, p_r\}$  forms a basis of the quotient space  $\mathbb{R}[x]/J$ .

Indeed if, for some scalars  $\lambda_i$ , the polynomial  $\sum_{i=1}^r \lambda_i p_i$  vanishes at all  $v_i$ 's, then  $\lambda_i = 0$  for all  $i$ . Hence the set  $\{p_1, \dots, p_r\}$  is linearly independent in  $\mathbb{R}[x]/J$  and thus it is a basis, since  $r = \dim \mathbb{R}[x]/J$ . Consider the linear form

$$\Lambda' := \sum_{i=1}^r \Lambda(p_i^2) \Lambda_{v_i}.$$

We claim that  $\Lambda = \Lambda'$ . As both  $\Lambda$  and  $\Lambda'$  vanish on the ideal  $J$ , it suffices to show that  $\Lambda$  and  $\Lambda'$  take the same values at all members of the basis  $\{p_1, \dots, p_r\}$  of  $\mathbb{R}[x]/J$ . Indeed,  $\Lambda'(p_j) = \Lambda(p_j^2)$  (since  $p_j(v_i) = \delta_{i,j}$ ), and  $\Lambda(p_j) = \Lambda(p_j^2)$  as well (since  $p_j - p_j^2 \in J$ ).  $\square$

*Example 2.6.* Consider the linear form  $\Lambda = \frac{1}{2}\Lambda_{(0,0)} + \frac{1}{2}\Lambda_{(1,2)} \in \mathbb{R}[x]^*$ , with moment matrix (indexed by  $1, x_1, x_2, x_1^2, \dots$ ):

$$M(\Lambda) = \begin{pmatrix} 1 & \frac{1}{2} & 1 & \frac{1}{2} & \dots \\ \frac{1}{2} & 1 & \frac{1}{2} & 1 & \dots \\ 1 & \frac{1}{2} & 1 & \frac{1}{2} & \dots \\ \frac{1}{2} & 1 & 2 & 1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} = \frac{1}{2}[v_1]_\infty[v_1]_\infty^T \Big|_{v_1=(0,0)} + \frac{1}{2}[v_2]_\infty[v_2]_\infty^T \Big|_{v_2=(1,2)}$$

Note e.g. that the 2nd and 4th columns of  $M(\Lambda)$  coincide, yielding the polynomial  $g_1 = -x_1 + x_1^2$  in the kernel of  $M(\Lambda)$ . In fact, the polynomials  $g_1, g_2 = -2x_1 + x_2, g_3 = -2x_1 + x_1x_2$  provide a basis of the real radical ideal  $\text{Ker } M(\Lambda)$ , whose variety is  $V_{\mathbb{C}}(\text{Ker } M(\Lambda)) = \{(0,0), (1,2)\} \subseteq \mathbb{R}^2$ .

As background information we mention (without proof) the following characterization for the linear forms  $\Lambda \in \mathbb{R}[x]^*$  with a finite rank moment matrix. When positivity is dropped, the evaluations at points  $v \in V_{\mathbb{C}}(\Lambda)$  do not suffice, one also needs the more general differential operators  $\partial_v^\alpha$  (defined in (2.4)).

**Theorem 2.7.** (see [9, Theorem 2.2.7], [10, Chap. 7]) Let  $\Lambda \in \mathbb{R}[x]^*$  satisfying  $\text{rank } M(\Lambda) < \infty$ . Say,  $V_{\mathbb{C}}(\text{Ker } M(\Lambda)) = \{v_1, \dots, v_r\}$ , so that  $r \leq \text{rank } M(\Lambda)$ . Then,

$$\Lambda = \sum_{i=1}^r \sum_{\alpha \in A_i} a_{\alpha,i} \partial_{v_i}^\alpha,$$

where  $A_i \subseteq \mathbb{N}^n$  are finite and  $a_{\alpha,i} \in \mathbb{R} \setminus \{0\}$ . Moreover,  $\text{Ker } M(\Lambda)$  is radical if and only if

$$\Lambda = \sum_{i=1}^r a_i \Lambda_{v_i}, \tag{2.8}$$

where  $a_i \neq 0$  (i.e.,  $A_i = \{0\}$  for all  $i$ ). Furthermore,  $\text{Ker } M(\Lambda)$  is real radical precisely when (2.8) holds with  $\{v_1, \dots, v_r\} \subseteq \mathbb{R}^n$ .

*Excursion:* Why is  $M(\Lambda)$  called a moment matrix? We briefly recall how the matrices  $M(\Lambda)$  arise naturally in the context of the classical moment problem in mathematics (cf. e.g. [1]). Given a finite positive Borel measure  $\mu$  on  $\mathbb{R}^n$ , the quantity

$$\int_{\mathbb{R}^n} x^\alpha d\mu$$

is called its *moment of order*  $\alpha \in \mathbb{N}^n$ , and the sequence  $y_\mu = (\int x^\alpha d\mu)_{\alpha \in \mathbb{N}^n}$  is called its *moment sequence*. The *moment problem* asks to characterize the sequences  $y \in \mathbb{R}^{\mathbb{N}^n}$

that are the sequence of moments of some finite positive Borel measure on (some subset of)  $\mathbb{R}^n$  or, equivalently, to characterize the linear forms  $\Lambda \in \mathbb{R}[x]^*$  of the form

$$\Lambda = \Lambda_\mu(p) := \int p(x)d\mu \text{ for } p \in \mathbb{R}[x]. \quad (2.9)$$

When (2.9) holds,  $\mu$  is called a *representing measure* for  $\Lambda$ . A well known result of Haviland [11] claims that  $\Lambda$  has a representing measure if and only if  $\Lambda(p) \geq 0$  for all polynomials  $p$  that are nonnegative on  $\mathbb{R}^n$ . However, except in some exceptional cases<sup>1</sup> no characterization is known for the nonnegative polynomials on  $\mathbb{R}^n$ . Yet we find the following well known necessary condition: If  $\Lambda$  has a representing measure, then  $\Lambda(p^2) \geq 0$  for all polynomials  $p$ , i.e.,  $\Lambda$  is positive, which is characterized by  $M(\Lambda) \geq 0$ .

Positivity of  $\Lambda$  is in general only a necessary condition for existence of a representing measure. However, the above result of Curto and Fialkow (Theorem 2.6) shows equivalence in the case when  $M(\Lambda)$  has finite rank, in which case the measure  $\mu$  is finite atomic with support  $V_{\mathbb{C}}(\text{Ker } M(\Lambda))$ .

When  $\mu = \delta_v$  is the Dirac measure at a point  $v \in \mathbb{R}^n$ , its moment sequence is  $y_\mu = [v]_\infty$  with corresponding linear form  $\Lambda_\mu = \Lambda_v$ , the evaluation at  $v$ . More generally, when  $\mu$  is finitely atomic, i.e., of the form  $\mu = \sum_{i=1}^r \lambda_i \delta_{v_i}$  with finite support  $\{v_1, \dots, v_r\} \subseteq \mathbb{R}^n$ , then its moment sequence is  $y_\mu = \sum_{i=1}^r \lambda_i [v_i]_\infty$  with corresponding linear form  $\Lambda_\mu = \sum_{i=1}^r \lambda_i \Lambda_{v_i}$ .

*Characterizing real radical ideals using positive linear forms on  $\mathbb{R}[x]$ .* We now combine the above results to obtain a semidefinite characterization of real radical ideals using positive linear forms. For this define the convex set

$$\mathcal{K} = \{\Lambda \in \mathbb{R}[x]^* \mid \Lambda(1) = 1, M(\Lambda) \geq 0 \text{ and } \Lambda(p) = 0 \forall p \in I\}. \quad (2.10)$$

For any  $\Lambda \in \mathcal{K}$ ,  $\text{Ker } M(\Lambda)$  is a real radical ideal, which contains  $I$  and thus its real radical  $\sqrt{\mathbb{R}I}$ . This implies

$$\dim \mathbb{R}[x]/\text{Ker } M(\Lambda) \leq \dim \mathbb{R}[x]/\sqrt{\mathbb{R}I}.$$

When the real variety  $V_{\mathbb{R}}(I)$  is finite,  $\mathbb{R}[x]/\sqrt{\mathbb{R}I}$  has finite dimension as a vector space, equal to  $|V_{\mathbb{R}}(I)|$ , and thus  $\text{Ker } M(\Lambda)$  is zero-dimensional with

$$\text{rank } M(\Lambda) = \dim \mathbb{R}[x]/\text{Ker } M(\Lambda) \leq \dim \mathbb{R}[x]/\sqrt{\mathbb{R}I} = |V_{\mathbb{R}}(I)|$$

---

<sup>1</sup>A celebrated result of Hilbert (cf. e.g. [2]) shows that there are three sets of parameters  $(n, d)$  for which the following equivalence holds: For any polynomial  $p$  in  $n$  variables and degree  $2d$ ,  $p$  is nonnegative on  $\mathbb{R}^n$  if and only if  $p$  can be written as a sum of squares of polynomials. These parameters are  $(n = 1, d)$  (univariate polynomials),  $(n, d = 1)$  (quadratic polynomials), and  $(n = 3, d = 2)$  (ternary quartic polynomials). In all other cases there are polynomials that are nonnegative on  $\mathbb{R}^n$  but cannot be written as a sum of squares of polynomials.

(using Lemma 2.2 for the left most equality). Equality:  $\text{rank } M(\Lambda) = |V_{\mathbb{R}}(I)|$  holds, for instance, for the element  $\Lambda = \frac{1}{|V_{\mathbb{R}}(I)|} \sum_{v \in V_{\mathbb{R}}(I)} \Lambda_v$  of  $\mathcal{K}$ . This fact motivates the following definition:

**Definition 2.4 (Generic linear forms).** Let  $\mathcal{K}$  be defined as in (2.10) and assume  $|V_{\mathbb{R}}(I)| < \infty$ . A linear form  $\Lambda \in \mathcal{K}$  is said to be *generic* if  $M(\Lambda)$  has maximum rank, i.e., if  $\text{rank } M(\Lambda) = |V_{\mathbb{R}}(I)|$ .

A simple geometric property of positive semidefinite matrices yields the following equivalent definition for generic elements of  $\mathcal{K}$ . This is in fact the key tool used in [17] for computing the real radical ideal  $\sqrt{\mathbb{R}I}$ .

**Lemma 2.4.** Assume  $|V_{\mathbb{R}}(I)| < \infty$ . An element  $\Lambda \in \mathcal{K}$  is generic if and only if  $\text{Ker } M(\Lambda) \subseteq \text{Ker } M(\Lambda')$  for all  $\Lambda' \in \mathcal{K}$ . Moreover,  $\text{Ker } M(\Lambda) = \sqrt{\mathbb{R}I}$  for all generic  $\Lambda \in \mathcal{K}$ .

*Proof.* Assume first that  $\text{rank } M(\Lambda) = r$ , with  $r = |V_{\mathbb{R}}(I)|$  and  $V_{\mathbb{R}}(I) = \{v_1, \dots, v_r\}$ . As  $\Lambda + \Lambda' \in \mathcal{K}$  for  $\Lambda' \in \mathcal{K}$ , we have

$$\text{Ker } M(\Lambda + \Lambda') = \text{Ker } M(\Lambda) \cap \text{Ker } M(\Lambda') \subseteq \text{Ker } M(\Lambda),$$

implying  $r \geq \text{rank } M(\Lambda + \Lambda') \geq \text{rank } M(\Lambda)$ . Hence equality holds throughout which implies  $\text{Ker } M(\Lambda) = \text{Ker } M(\Lambda) \cap \text{Ker } M(\Lambda') \subseteq \text{Ker } M(\Lambda')$ .

Conversely, assume  $\text{Ker } M(\Lambda) \subseteq \text{Ker } M(\Lambda')$  for all  $\Lambda' \in \mathcal{K}$ . Consider  $\Lambda' = \sum_{i=1}^r \Lambda_{v_i} \in \mathcal{K}$  whose kernel is  $I(v_1, \dots, v_r)$ . This implies  $\text{Ker } M(\Lambda) \subseteq I(v_1, \dots, v_r)$  and thus

$$\text{rank } M(\Lambda) = \dim \mathbb{R}[x]/\text{Ker } M(\Lambda) \geq \dim \mathbb{R}[x]/I(v_1, \dots, v_r) = r.$$

Hence,  $\text{rank } M(\Lambda) = r$  and  $\text{Ker } M(\Lambda) = I(v_1, \dots, v_r) = \sqrt{\mathbb{R}I}$  (using the Real Nullstellensatz, Theorem 2.1 (ii), for the last equality).  $\square$

*Example 2.7 (Example 2.6 cont.).* Consider the set  $\mathcal{K}$  corresponding to the ideal  $I = \langle h_1, h_2, h_3 \rangle \subseteq \mathbb{R}[x_1, x_2]$ , where

$$h_1 = x_2^4 x_1 + 3x_2^3 - x_2^4 - 3x_1^2, \quad h_2 = x_1^2 x_2 - 2x_1^2, \quad h_3 = 2x_2^4 x_1 - x_1^3 - 2x_2^4 + x_1^2.$$

Then,  $\Lambda = \frac{1}{2}\Lambda_{(0,0)} + \frac{1}{2}\Lambda_{(1,2)}$  is a generic element of  $\mathcal{K}$ . Thus the real radical ideal of  $I$  is  $\sqrt{\mathbb{R}I} = \text{Ker } M(\Lambda) = \langle g_1, g_2, g_3 \rangle$ , with  $g_1, g_2, g_3$  as in Example 2.6.

### 2.3.2 Truncated Positive Linear Forms and Real Radical Ideals

In view of the results in the previous section (in particular, Lemmas 2.2 and 2.4), the task of finding the real radical ideal  $\sqrt{\mathbb{R}I}$  as well as a linear basis of the quotient space

$\mathbb{R}[x]/\sqrt{I}$  can be reduced to finding a generic linear form  $\Lambda$  in the set  $\mathcal{K}$  (defined in (2.10)). In order to be able to deal with such linear forms computationally, we will work with linear forms on finite dimensional subspaces  $\mathbb{R}[x]_s$  of the polynomial ring. Given  $\Lambda \in (\mathbb{R}[x]_{2s})^*$ , we can define the quadratic form:

$$Q_\Lambda : f \in \mathbb{R}[x]_s \mapsto Q_\Lambda(f) = \Lambda(f^2),$$

whose matrix

$$M_s(\Lambda) = (\Lambda(x^\alpha x^\beta))_{\alpha, \beta \in \mathbb{N}_s^n}$$

in the monomial basis of  $\mathbb{R}[x]_s$  is called the *truncated moment matrix of order s of  $\Lambda$* . Thus  $\Lambda$  is positive (i.e.,  $\Lambda(f^2) \geq 0 \quad \forall f \in \mathbb{R}[x]_s$ ) if and only if  $M_s(\Lambda) \succeq 0$ . Again we identify the kernels of  $Q_\Lambda$  and of  $M_s(\Lambda)$  (by identifying polynomials with their coefficient sequences) and view  $\text{Ker } M_s(\Lambda)$  as a subset of  $\mathbb{R}[x]_s$ .

*Flat extensions of moment matrices.* We now present the following crucial result of Curto and Fialkow [7] for flat extensions of moment matrices.

**Theorem 2.8 (Flat extension theorem [7], see also [21]).** *Let  $\Lambda \in (\mathbb{R}[x]_{2s})^*$  and assume that  $M_s(\Lambda)$  is a flat extension of  $M_{s-1}(\Lambda)$ , i.e.,*

$$\text{rank } M_s(\Lambda) = \text{rank } M_{s-1}(\Lambda). \tag{2.11}$$

*Then one can extend (uniquely)  $\Lambda$  to  $\tilde{\Lambda} \in (\mathbb{R}[x]_{2s+2})^*$  in such a way that  $M_{s+1}(\tilde{\Lambda})$  is a flat extension of  $M_s(\Lambda)$ ; thus  $\text{rank } M_{s+1}(\tilde{\Lambda}) = \text{rank } M_s(\Lambda)$ .*

The proof is elementary and relies on the following lemma showing that the kernel of a truncated moment matrix behaves like a ‘truncated ideal’.

**Lemma 2.5.** *Let  $\Lambda \in (\mathbb{R}[x]_{2s})^*$  and  $f, g \in \mathbb{R}[x]$  with  $f \in \text{Ker } M_s(\Lambda)$ .*

- (i) *Assume  $\text{rank } M_s(\Lambda) = \text{rank } M_{s-1}(\Lambda)$ . Then  $\text{Ker } M_{s-1}(\Lambda) \subseteq \text{Ker } M_s(\Lambda)$  and  $fg \in \text{Ker } M_s(\Lambda)$  if  $\deg(fg) \leq s$ .*
- (ii) *Assume  $M_s(\Lambda) \succeq 0$ . Then  $\text{Ker } M_{s-1}(\Lambda) \subseteq \text{Ker } M_s(\Lambda)$  and  $fg \in \text{Ker } M_s(\Lambda)$  if  $\deg(fg) \leq s-1$ .*

Indeed, using property (2.11) and Lemma 2.5 (i), we see that for every monomial  $m$  of degree  $s$ , there exists a polynomial of the form  $f_m = m + r_m \in \text{Ker } M_s(\Lambda)$ , where  $r_m \in \mathbb{R}[x]_{s-1}$ . If an extension  $\tilde{\Lambda}$  exists, then all the polynomials  $f_m, x_i f_m$  must lie in the kernel of  $M_{s+1}(\tilde{\Lambda})$  and they can be used to determine the unknown columns of  $M_{s+1}(\tilde{\Lambda})$  indexed by monomials of degree  $s+1$ . The main work consists of verifying the consistency of this construction; namely, that the matrix constructed in this way is a moment matrix, i.e. that its  $(\alpha, \beta)$ th entry depends only on the sum  $\alpha + \beta$  when  $|\alpha + \beta| = 2s+1, 2s+2$ .

The flat extension theorem plays a crucial role in the moment matrix approach as it allows to deduce information about the infinite moment matrix  $M(\Lambda)$  from its finite section  $M_s(\Lambda)$ .

**Theorem 2.9 ([17]).** Let  $\Lambda \in (\mathbb{R}[x]_{2S})^*$  and assume that (2.11) holds. Then one can extend  $\Lambda$  to  $\tilde{\Lambda} \in \mathbb{R}[x]^*$  in such a way that  $M(\tilde{\Lambda})$  is a flat extension of  $M_s(\Lambda)$ , and the ideal  $\text{Ker } M(\tilde{\Lambda})$  is generated by the polynomials in  $\text{Ker } M_s(\Lambda)$ , i.e.,

$$\text{rank } M(\tilde{\Lambda}) = \text{rank } M_s(\Lambda) \text{ and } \text{Ker } M(\tilde{\Lambda}) = \langle \text{Ker } M_s(\Lambda) \rangle.$$

Moreover, any monomial set  $\mathcal{B}$  indexing a basis of the column space of  $M_{s-1}(\Lambda)$  is a basis of the quotient space  $\mathbb{R}[x]/\text{Ker } M(\tilde{\Lambda})$ . If, moreover,  $M_s(\Lambda) \succeq 0$ , then the ideal  $\langle \text{Ker } M_s(\Lambda) \rangle$  is real radical and  $\Lambda$  is of the form  $\Lambda = \sum_{i=1}^r \lambda_i \Lambda_{v_i}$ , where  $\lambda_i > 0$  and  $\{v_1, \dots, v_r\} = V_{\mathbb{C}}(\text{Ker } M_s(\Lambda)) \subseteq \mathbb{R}^n$ .

*Proof.* The existence of  $\tilde{\Lambda}$  follows by applying iteratively Theorem 2.8 and the inclusion  $\langle \text{Ker } M_s(\Lambda) \rangle \subseteq \text{Ker } M(\tilde{\Lambda})$  follows using Lemma 2.5 (i). If  $\mathcal{B}$  is a set of monomials indexing a column basis of  $M_{s-1}(\Lambda)$ , then  $\mathcal{B}$  is also a column basis of  $M(\tilde{\Lambda})$  and thus a basis of  $\mathbb{R}[x]/\text{Ker } M(\tilde{\Lambda})$  (by Lemma 2.2). One can verify the direct sum decomposition  $\mathbb{R}[x] = \text{Span}_{\mathbb{R}}(\mathcal{B}) \oplus \langle \text{Ker } M_s(\Lambda) \rangle$ , which implies  $\text{Ker } M(\tilde{\Lambda}) = \langle \text{Ker } M_s(\Lambda) \rangle$ . Finally, as  $\tilde{\Lambda}$  is a flat extension of  $\Lambda$ ,  $M_s(\Lambda) \succeq 0$  implies  $M(\tilde{\Lambda}) \succeq 0$ , so that  $\langle \text{Ker } M_s(\Lambda) \rangle = \text{Ker } M(\tilde{\Lambda})$  is real radical (by Lemma 2.1). The final statement follows directly by applying Theorem 2.6 to  $\tilde{\Lambda}$ .  $\square$

*Example 2.8 (Example 2.6 cont).* Consider the linear form  $\Lambda \in \mathbb{R}[x]^*$  in Example 2.6. Recall that  $\text{Ker } M(\Lambda)$  is generated by  $g_1, g_2, g_3 \in \mathbb{R}[x]_2$ . First note that these polynomials imply the rank condition:  $\text{rank } M_2(\Lambda) = \text{rank } M_1(\Lambda)$  and thus permit to construct  $M_2(\Lambda)$  from  $M_1(\Lambda)$ . Moreover, they permit to recover the infinite matrix  $M(\Lambda)$  from its submatrix  $M_1(\Lambda)$ . For instance, since  $x_1^2 x_2 = x_2(x_1 + g_1) = 2x_1 + g_3 + g_1 x_2$  and  $g_1, g_2, g_3 \in \text{Ker } M_2(\Lambda) \subseteq \text{Ker } M(\Lambda)$ , we deduce that the column of  $M(\Lambda)$  indexed by  $x_1^2 x_2$  is equal to twice its column indexed by  $x_1$ . Using the fact that  $\text{Ker } M(\Lambda) = \langle \text{Ker } M_2(\Lambda) \rangle$ , we can analogously define iteratively all columns of  $M(\Lambda)$ .

*Computing real radical ideals using truncated positive linear forms on  $\mathbb{R}[x]_t$ .* We saw above how to use positive linear forms on  $\mathbb{R}[x]$  to characterize the real radical ideal  $\sqrt[2]{I}$ . We now combine this characterization with the above results about flat extensions of truncated moment matrices to obtain a practical algorithm for computing  $\sqrt[2]{I}$  operating on finite dimensional subspaces  $\mathbb{R}[x]_t \subseteq \mathbb{R}[x]$  only. As before  $I = \langle h_1, \dots, h_m \rangle$  is the ideal generated by the polynomial equations  $h_i$  to be solved. For  $t \in \mathbb{N}$ , define the set

$$\mathcal{H}_t = \{h_i x^\alpha \mid i = 1, \dots, m, |\alpha| \leq t - \deg(h_i)\} \quad (2.12)$$

of prolongations up to degree  $t$  of the polynomials  $h_i$ , and the truncated analogue of the set  $\mathcal{K}$ :

$$\mathcal{K}_t = \{\Lambda \in (\mathbb{R}[x]_t)^* \mid \Lambda(1) = 1, M_{[t/2]}(\Lambda) \succeq 0 \text{ and } \Lambda(f) = 0 \ \forall f \in \mathcal{H}_t\}. \quad (2.13)$$

Note that the constraint:  $\Lambda(f) = 0 \ \forall f \in \mathcal{H}_t$  (i.e.,  $\Lambda \in \mathcal{H}_t^\perp$ ) corresponds to the constraint (2.2) of Sect. 2.1.2. As the convex set  $\mathcal{K}_t$  is described by the positive

semidefiniteness of an affinely parametrized matrix, it is an instance of a *spectrahedron*, cf. Chaps. 5 and 13 of this volume. The following lemma is the truncated analogue of Lemma 2.4.

**Lemma 2.6 (Generic truncated linear forms).** *The following assertions are equivalent for  $\Lambda \in (\mathbb{R}[x]_t)^*$ :*

- (i)  $\text{rank } M_{\lfloor t/2 \rfloor}(\Lambda) \geq \text{rank } M_{\lfloor t/2 \rfloor}(\Lambda')$  for all  $\Lambda' \in \mathcal{K}_t$ .
- (ii)  $\text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda) \subseteq \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda')$  for all  $\Lambda' \in \mathcal{K}_t$ .
- (iii) *The linear form  $\Lambda$  lies in the relative interior of the convex set  $\mathcal{K}_t$ .*

*Then  $\Lambda$  is called a generic element of  $\mathcal{K}_t$  and the kernel  $\mathcal{N}_t = \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$  is independent of the particular choice of the generic element  $\Lambda \in \mathcal{K}_t$ .*

**Theorem 2.10.** *We have:  $\mathcal{N}_t \subseteq \mathcal{N}_{t+1} \subseteq \dots \subseteq \sqrt[t]{I}$ , with equality  $\sqrt[t]{I} = \langle \mathcal{N}_t \rangle$  for  $t$  large enough.*

*Proof.* Let  $\Lambda \in \mathcal{K}_{t+1}$  be generic. Its restriction to  $(\mathbb{R}[x]_t)^*$  lies in  $\mathcal{K}_t$ , implying

$$\mathcal{N}_{t+1} = \text{Ker } M_{\lfloor (t+1)/2 \rfloor}(\Lambda) \supseteq \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda) \supseteq \mathcal{N}_t.$$

Now let  $\Lambda$  be a generic element of  $\mathcal{K}_t$  so that  $\mathcal{N}_t = \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$ . The inclusion:  $\mathcal{N}_t \subseteq \mathcal{I}(V_{\mathbb{R}}(I))$  follows using Lemma 2.6 (ii). Indeed,  $\Lambda_v \in \mathcal{K}_t$  for all  $v \in V_{\mathbb{R}}(I)$ , which implies  $\text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda) \subseteq \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda_v) \subseteq \mathcal{I}(v)$  and thus  $\text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda) \subseteq \bigcap_{v \in V_{\mathbb{R}}(I)} \mathcal{I}(v) = \mathcal{I}(V_{\mathbb{R}}(I)) = \sqrt[t]{I}$  (by the Real Nullstellensatz).

We now show equality:  $\sqrt[t]{I} = \langle \mathcal{N}_t \rangle$  for  $t$  large enough. For this, let  $\{g_1, \dots, g_L\}$  be a basis of the ideal  $\sqrt[t]{I}$ ; we show that  $g_l \in \mathcal{N}_t$  for all  $l$ . We have:

$$g_l^{2k} + \sum_j s_j^2 = \sum_{i=1}^m u_i h_i \quad \text{for some } k \in \mathbb{N} \text{ and } s_j, u_i \in \mathbb{R}[x].$$

Since  $\Lambda \in \mathcal{H}_t^\perp$ , we have  $h_i \in \mathcal{N}_t$  if  $t \geq 2 \deg(h_i)$ . Using Lemma 2.5 (ii), this implies that, for  $t$  large enough,  $\mathcal{N}_t$  contains each  $u_i h_i$  and thus  $g_l^{2k} + \sum_j s_j^2$ . In particular,  $\Lambda(g_l^{2k} + \sum_j s_j^2) = 0$ . On the other hand,  $\Lambda(g_l^{2k}), \Lambda(s_j^2) \geq 0$  (since  $M_{\lfloor t/2 \rfloor}(\Lambda) \geq 0$ ), thus implying  $\Lambda(g_l^{2k}) = 0$ . An easy induction on  $k$  now permits to conclude that  $g_l \in \mathcal{N}_t$ .  $\square$

When  $V_{\mathbb{R}}(I)$  is finite, one can guaranty the equality  $\sqrt[t]{I} = \langle \mathcal{N}_t \rangle$  using the rank condition (2.11). The next results provide all the ingredients of the moment matrix algorithm for real roots, whose description is given in Sect. 2.3.3: Theorem 2.11 will provide a stopping criterion (when  $|V_{\mathbb{R}}(I)| < \infty$ ) and Theorem 2.12 below will imply its termination, as well as provide a criterion permitting to check the (non-)existence of real roots.

**Theorem 2.11 ([17]).** *Let  $I = \langle h_1, \dots, h_m \rangle$  be an ideal in  $\mathbb{R}[x]$ ,  $D = \max_i \deg(h_i)$ , and  $d = \lceil D/2 \rceil$ . Let  $\Lambda \in \mathcal{K}_t$  be a generic element and assume that at least one of the*

following two conditions holds:

$$\text{rank } M_s(\Lambda) = \text{rank } M_{s-1}(\Lambda) \quad \text{for some } D \leq s \leq \lfloor t/2 \rfloor, \quad (2.14)$$

$$\text{rank } M_s(\Lambda) = \text{rank } M_{s-d}(\Lambda) \quad \text{for some } d \leq s \leq \lfloor t/2 \rfloor. \quad (2.15)$$

Then,  $\sqrt[{\mathbb{R}}]{I} = \langle \text{Ker } M_s(\Lambda) \rangle$ , and any basis of the column space of  $M_{s-1}(\Lambda)$  is a basis of the quotient space  $\mathbb{R}[x]/\sqrt[{\mathbb{R}}]{I}$ .

*Proof.* The ideal  $J := \langle \text{Ker } M_s(\Lambda) \rangle$  is real radical (by Theorem 2.9). Moreover,

$$\text{Ker } M_s(\Lambda) \subseteq \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda) \subseteq \sqrt[{\mathbb{R}}]{I}$$

(since  $\Lambda$  is generic and using Theorem 2.10) and thus  $J \subseteq \sqrt[{\mathbb{R}}]{I}$ . Remains to show  $\sqrt[{\mathbb{R}}]{I} \subseteq J$ . Suppose first that (2.14) holds. The condition  $\Lambda \in \mathcal{H}_t^\perp$  implies that  $h_i \in \text{Ker } M_s(\Lambda)$  (since  $s + \deg(h_i) \leq t/2 + \deg(h_i) \leq t$ , as  $t \geq 2D$ ). Thus  $I \subseteq J$ , implying  $\sqrt[{\mathbb{R}}]{I} \subseteq J$  as  $J$  is real radical.

Suppose now that (2.15) holds. Again from Theorem 2.9 we know that  $V_{\mathbb{C}}(\text{Ker } M_s(\Lambda)) = \{v_1, \dots, v_r\} \subseteq \mathbb{R}^n$  and  $\Lambda = \sum_{i=1}^r \lambda_i \Lambda_{v_i}$  where  $\lambda_i > 0$ . Let  $p_1, \dots, p_r$  be interpolation polynomials at the  $v_i$ 's, i.e., such that  $p_j(v_i) = \delta_{i,j}$ . An easy but crucial observation (made in [20]) is that we may assume that each  $p_j$  has degree at most  $s-d$ . Indeed, we can replace each interpolation polynomial  $p_j$  by its normal form modulo  $J$  with respect to a basis of  $\mathbb{R}[x]/J$ . As such a basis can be obtained by picking a column basis of  $M_{s-d}(\Lambda)$ , its members are monomials of degree at most  $s-d$ , and the resulting normal forms of the  $p_j$ 's are again interpolation polynomials at the  $v_i$ 's but now with degree at most  $s-d$ . As  $\deg(p_j^2) \leq 2(s-d) \leq t-2d \leq t-\deg(h_i)$ , we can claim that  $\Lambda(p_j^2 h_i) = 0$  and in turn  $0 = \Lambda(p_j^2 h_i) = \sum_{l=1}^r \lambda_l p_j^2(v_l) h_i(v_l) = \lambda_j h_i(v_j)$ . Since  $h_i(v_j) = 0$  for all  $i, j$ , we conclude that  $\{v_1, \dots, v_r\} \subseteq V_{\mathbb{R}}(I)$ , implying the desired inclusion  $\sqrt[{\mathbb{R}}]{I} = I(V_{\mathbb{R}}(I)) \subseteq I(v_1, \dots, v_r) = J$ .  $\square$

**Theorem 2.12 ([17]).** *Let  $I$  be an ideal in  $\mathbb{R}[x]$ .*

- (i) *If  $V_{\mathbb{R}}(I) = \emptyset$ , then  $\mathcal{K}_t = \emptyset$  for  $t$  large enough.*
- (ii) *If  $1 \leq |V_{\mathbb{R}}(I)| < \infty$  then, for  $t$  large enough, there exists an integer  $s$  for which (2.15) holds for all  $\Lambda \in \mathcal{K}_t$ .*

*Proof.* Let  $\{g_1, \dots, g_L\}$  be a Gröbner basis of  $\sqrt[{\mathbb{R}}]{I}$  with respect to a total degree monomial ordering, and let  $\mathcal{B}$  be the corresponding set of standard monomials, forming a basis of  $\mathbb{R}[x]/\sqrt[{\mathbb{R}}]{I}$ . The argument used in the proof of Theorem 2.10 shows the existence of  $t_0 \in \mathbb{N}$  for which  $\{g_1, \dots, g_L\} \subseteq \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$  for all  $t \geq t_0$  and  $\Lambda \in \mathcal{K}_t$ .

- (i) If  $V_{\mathbb{R}}(I) = \emptyset$ , then  $\{1\}$  is a basis of  $\sqrt[{\mathbb{R}}]{I} = \mathbb{R}[x]$ . Thus  $1 \in \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$ , implying  $\Lambda(1) = 0$  if  $\Lambda \in \mathcal{K}_t$ , contradicting  $\Lambda(1) = 1$  and thus showing  $\mathcal{K}_t = \emptyset$ .

**Table 2.1** Ranks of  $M_s(\Lambda)$  for generic  $\Lambda \in \mathcal{K}_t$  in Example 2.9

$s =$	0	1	2	3
$t = 5$	1	3	5	–
$t = 6$	1	<b>2</b>	<b>2</b>	4

- (ii) As  $V_{\mathbb{R}}(I)$  is finite,  $s := d + \max_{b \in \mathcal{B}} \deg(b)$  is well defined. Recall that  $d = \max_i \lceil \deg(h_i)/2 \rceil$ . Choose  $t \geq t_0$  such that  $s < \lfloor t/2 \rfloor$ . For  $\alpha \in \mathbb{N}_s^n$ , decompose  $x^\alpha$  as

$$x^\alpha = \sum_{b \in \mathcal{B}} \lambda_b b + \sum_{l=1}^L u_l g_l \in \text{Span}_{\mathbb{R}}(\mathcal{B}) \oplus \sqrt[3]{I},$$

where  $\lambda_b \in \mathbb{R}$ ,  $u_l \in \mathbb{R}[x]$ ,  $\deg(\sum_b \lambda_b b) \leq s-d$ , and  $\deg(u_l g_l) \leq s < \lfloor t/2 \rfloor$  (as the  $g_l$ 's form a Gröbner basis for a *total degree ordering*, we can claim  $\deg(u_l g_l) \leq s$ ). As  $g_l \in \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$  and  $\deg(u_l g_l) < \lfloor t/2 \rfloor$ , we also have that  $u_l g_l \in \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$  (recall Lemma 2.5 (ii)). Hence,  $x^\alpha - \sum_{b \in \mathcal{B}} \lambda_b b \in \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$ , which shows that  $M_s(\Lambda)$  is a flat extension of  $M_{s-d}(\Lambda)$ .  $\square$

*Example 2.9 (Example 2.7 cont).* Consider again the ideal  $I = (h_1, h_2, h_3)$  from Example 2.7. Then,  $D = 5$ ,  $\dim \mathbb{R}[x]/I = 9$ , and the variety  $V_{\mathbb{C}}(I)$  consists of two real points, one of them with multiplicity eight. Table 2.1 shows the ranks of the moment matrix  $M_s(\Lambda)$  for generic  $\Lambda \in \mathcal{K}_t$ . The rank condition holds at order  $(t, s) = (6, 2)$ . Then we can extract the two roots  $v_1 = (0, 0)$  and  $v_2 = (1, 2)$  as well as the (border) basis  $\{g_1, g_2, g_3\}$  of  $\sqrt[3]{I}$  (already discussed in Example 2.7). This is possible although here  $s=2$  is strictly smaller than  $d=3$  and  $D=5$ ; indeed, in view of Theorem 2.5, we can simply check whether the formal multiplication matrices commute and whether  $h_i(v) = 0$  for all  $i = 1, \dots, m$  and  $v \in V_{\mathbb{C}}(\text{Ker } M_s(\Lambda))$ .

We conclude with two remarks about Theorem 2.11, which is the central result of this section. Namely we mention a generalization and an application.

First, observe that one may work with moment matrices  $M_{\mathcal{B}}(y)$  indexed by an arbitrary monomial set  $\mathcal{B}$ , instead of moment matrices  $M_t(y)$  indexed by all monomials up to a given degree  $t$ , which leads to possible generalizations of Theorem 2.11. More precisely, let  $\Lambda$  be a generic element in  $\mathcal{K}_t$ . Assume that we can find a monomial set  $\mathcal{B}$ , connected to 1, indexing a linearly independent set of columns of the moment matrix  $M_{\lfloor t/2 \rfloor}(\Lambda)$ , and for which the submatrices  $M_{\mathcal{B}}(\Lambda)$  and  $M_{\mathcal{B}^+}(\Lambda)$  indexed, respectively, by  $\mathcal{B}$  and  $\mathcal{B}^+$ , satisfy the rank condition:

$$\text{rank } M_{\mathcal{B}^+}(\Lambda) = \text{rank } M_{\mathcal{B}}(\Lambda).$$

Then one can show that the ideal  $J = \langle \text{Ker } M_{\mathcal{B}^+}(\Lambda) \rangle$  is real radical, zero-dimensional, and contained in  $\sqrt[3]{I}$ , and thus  $V_{\mathbb{R}}(I) \subseteq V_{\mathbb{C}}(J)$ ; this result relies on a generalization of the flat extension theorem (Theorem 2.8) proved in [22, Theorem 1.4]. Hence, one can compute the variety  $V_{\mathbb{C}}(J) \subseteq \mathbb{R}^n$ , and select from it the desired real variety  $V_{\mathbb{R}}(I)$ .

Next, as a byproduct of Theorem 2.11, we see that the rank condition (2.14) (or (2.15)) also implies a full description of the convex hull of the variety  $V_{\mathbb{R}}(I)$ . Indeed, under (2.14), we can apply Theorem 2.9 to deduce that, for any  $\Lambda \in \mathcal{K}_t$ , its restriction  $\pi_{2s}(\Lambda)$  can be written as a conic combination of evaluations at points of  $V_{\mathbb{R}}(I)$ . Combining with Theorem 2.12, we obtain:

**Corollary 2.1.** *Assume  $1 \leq |V_{\mathbb{R}}(I)| < \infty$ . For some integers  $1 \leq s \leq \lfloor t/2 \rfloor$ , the set*

$$\{(\Lambda(x^\alpha))_{\alpha \in \mathbb{N}_{2s}^n} \mid \Lambda \in \mathcal{K}_t\}$$

*is equal to the convex hull of the set  $\{[v]_{2s} \mid v \in V_{\mathbb{R}}(I)\}$ .*

The Chap. 5 of this Handbook considers in detail the problem of characterizing the convex hull of a real variety  $V_{\mathbb{R}}(I)$ . Although the points of view and emphasis are different in both chapters, there are some connections. Roughly speaking, both chapters can be cast within the more general realm of polynomial optimization (see Sect. 2.4.1); however, while we work here with truncated sections of the ideal  $I$ , Chap. 5 deals with linear forms on the full quotient space  $\mathbb{R}[x]/I$ .

### 2.3.3 The Moment Matrix Algorithm for Computing Real Roots

We now describe the moment matrix algorithm for computing real roots, summarized in Algorithm 1 below.

---

#### Algorithm 1 The moment matrix algorithm for $V_{\mathbb{R}}(I)$

---

**Input:** Generators  $h_1, \dots, h_m$  of some ideal  $I = \langle h_1, \dots, h_m \rangle$  with  $|V_{\mathbb{R}}(I)| < \infty$ .  
**Output:** A basis of the ideal  $\sqrt{\mathbb{R}I}$ , a basis of  $\mathbb{R}[x]/\sqrt{\mathbb{R}I}$ , and the set  $V_{\mathbb{R}}(I)$ .

- 1: Set  $t = D$ .
- 2: Find a generic element  $\Lambda \in \mathcal{K}_t$ .
- 3: Check if (2.14) holds for some  $D \leq s \leq \lfloor t/2 \rfloor$ ,  
or if (2.15) holds for some  $d \leq s \leq \lfloor t/2 \rfloor$ .
- 4: **if** yes **then**
- 5:   Set  $J = \langle \text{Ker } M_s(\Lambda) \rangle$ .
- 6:   Compute a basis  $\mathcal{B} \subseteq \mathbb{R}[x]_{s-1}$  of the column space of  $M_{s-1}(\Lambda)$ .
- 7:   Compute the multiplication matrices  $X_i$  in  $\mathbb{R}[x]/J$ .
- 8:   Compute a basis  $g_1, \dots, g_l \in \mathbb{R}[x]_s$  of the ideal  $J$ .
- 9:
- 10:   **return** the basis  $\mathcal{B}$  of  $\mathbb{R}[x]/J$  and the generators  $g_1, \dots, g_l$  of  $J$ .
- 11: **else**
- 12:   Iterate (go to Step 2) replacing  $t$  by  $t + 1$ .
- 13: **end if**
- 14: Compute  $V_{\mathbb{R}}(I) = V_{\mathbb{C}}(J)$  (via the eigenvalues/eigenvectors of the multiplication matrices  $X_i$ ).

---

Theorem 2.11 implies the correctness of the algorithm (i.e., equality  $J = \sqrt[|\mathbb{R}|]{I}$ ) and Theorem 2.12 shows its termination. Algorithm 1 consists of four main parts, which we now briefly discuss (see [17, 35] for details).

- (i) *Finding a generic element in  $\mathcal{K}_t$ .* The set  $\mathcal{K}_t$  can be represented as the feasible region of a semidefinite program and we have to find a point lying in its relative interior. Such a point can be found by solving several semidefinite programs with an arbitrary SDP solver (cf. [17, Remark 4.15]), or by solving a single semidefinite program with an interior-point algorithm using a self-dual embedding technique (see, e.g., [8, 44]). Indeed consider the semidefinite program:

$$\begin{aligned} \min_{\Lambda \in (\mathbb{R}[x])_t^*} \quad & 1 \text{ such that } \Lambda(1) = 1, M_{\lfloor t/2 \rfloor}(\Lambda) \succeq 0, \\ & \Lambda(h_i x^\alpha) = 0 \quad \forall i \quad \forall |\alpha| \leq t - \deg(h_i), \end{aligned} \quad (2.16)$$

whose dual reads:

$$\begin{aligned} \max \lambda \text{ such that } \quad & 1 - \lambda = s + \sum_{i=1}^m u_i h_i \text{ where } s, u_i \in \mathbb{R}[x], \\ & s \text{ is a sum of squares, } \deg(s), \deg(u_i h_i) \leq t. \end{aligned} \quad (2.17)$$

The feasible region of (2.16) is the set  $\mathcal{K}_t$ , as well as its set of optimal solutions, since we minimize a constant objective function over  $\mathcal{K}_t$ . There is no duality gap, as  $\lambda = 1$  is obviously feasible for (2.17). Solving the program (2.16) with an interior-point algorithm using a self-dual embedding technique yields<sup>2</sup> either a solution  $\Lambda$  lying in the relative interior of the optimal face (i.e., a generic element of  $\mathcal{K}_t$ ), or a certificate that (2.16) is infeasible thus showing  $V_{\mathbb{R}}(I) = \emptyset$ .

- (ii) *Computing the ranks of submatrices of  $M_t(\Lambda)$ .* In order to check whether one of the conditions (2.14) or (2.15) holds we need to compute the ranks of matrices consisting of numerical values. This computationally challenging task may be done by detecting zero singular values and/or a large decay between two subsequent values.
- (iii) *Computing a basis  $\mathcal{B}$  for the column space of  $M_{s-1}(\Lambda)$ .* The set of monomials  $\mathcal{B}$  indexing a maximum nonsingular principle submatrix of  $M_s(\Lambda)$  directly reveals a basis of the quotient space  $\mathbb{R}[x]/J$  (by Theorem 2.9). The choice of this basis may influence the numerical stability of the extracted set of solutions and the properties of the border basis of  $J$  as well. The options range from a monomial basis obtained using a greedy algorithm or more sophisticated polynomial bases (see [35]).

---

<sup>2</sup>This follows under certain technical conditions on the semidefinite program, which are satisfied for (2.16); see [17] for details.

- (iv) *Computing a basis of  $J$  and the formal multiplication matrices.* Say  $\mathcal{B}$  is the monomial basis (connected to 1) of the column space of  $M_{s-1}(\Lambda)$  constructed at the previous step (iii). Under the rank condition (2.14) or (2.15), for any  $b \in \mathcal{B}$ , the monomial  $x_i b$  can be written as  $x_i b = r_{i,b} + q$ , where  $r_{i,b} \in \text{Span}_{\mathbb{R}}(\mathcal{B})$  and  $q \in \text{Ker } M_s(\Lambda)$ . These polynomials directly give a (border) basis of  $J$ , consisting of the polynomials  $\{x_i b - r_{i,b} \mid i \leq n, b \in \mathcal{B}\}$  (recall Theorem 2.5) and thus permit the construction of multiplication matrices and the computation of  $V_{\mathbb{C}}(J)$  ( $= V_{\mathbb{R}}(I)$ ).

*Existing implementations and performance.* The basic algorithm discussed above has been implemented in Matlab using Yalmip (see [25]) as part of a new toolbox *Bermeja* for computations in Convex Algebraic Geometry (see [36]). In its current form, the implemented algorithm merely provides a proof of concept and only solves real root finding problems with a rather limited number of variables ( $\leq 10$ ) and of moderate degree ( $\leq 6$ ). This is mainly due to the fact that sparsity in the support of the polynomials is not utilized. This leads to large moment matrices, easily touching on the limitations of current SDP solvers. We refer to [17, 19] for a more detailed discussion and some numerical results. In an ongoing project a more efficient, Buchberger-style, version of this real root finding method will be implemented based on the more general version of the flat extension theorem, which was described at the end of Sect. 2.3.2. A flavor of how existing complex root finding methods may be tailored for real root finding is discussed in the next section.

### 2.3.4 Real vs. Complex Root Finding

As we saw in the previous section, the moment matrix approach for real roots relies on finding a suitable (generic) linear form  $\Lambda$  in the convex set  $\mathcal{K}_t$  (from (2.13)). Let us stress again that the positivity condition on  $\Lambda$  is the essential ingredient that permits to focus solely on the real roots among the complex ones. This is best illustrated by observing (following [18]) that, if we delete the positivity condition in the moment matrix algorithm (Algorithm 1), then the *same* algorithm permits to compute all *complex* roots (assuming their number is finite). In other words, consider the following analogue of the set  $\mathcal{K}_t$ :

$$\mathcal{K}_t^{\mathbb{C}} = \{\Lambda \in (\mathbb{R}[x]_t)^* \mid \Lambda(1) = 1 \text{ and } \Lambda(f) = 0 \ \forall f \in \mathcal{H}_t\}, \quad (2.18)$$

where  $\mathcal{H}_t$  is as in (2.12). Call an element  $\Lambda \in \mathcal{K}_t^{\mathbb{C}}$  *generic*<sup>3</sup> if  $\text{rank } M_s(\Lambda)$  is maximum for all  $s \leq \lfloor t/2 \rfloor$ . Then the moment matrix algorithm for complex roots is analogous to Algorithm 1, but with the following small twist: Instead of computing a generic element in the convex set  $\mathcal{K}_t$ , we have to compute a generic (aka random) element in

---

<sup>3</sup>When  $\Lambda$  is positive, the maximality condition on the rank of  $M_{\lfloor t/2 \rfloor}(\Lambda)$  implies that the rank of  $M_s(\Lambda)$  is maximum for all  $s \leq \lfloor t/2 \rfloor$ . This is not true for  $\Lambda$  non-positive.

**Table 2.2** Ranks of  $M_s(\Lambda)$  in Example 2.10

(a) Generic $\Lambda \in \mathcal{K}_t$				(b) Generic $\Lambda \in \mathcal{H}_t^\perp$						
$s =$	0	1	2	3	$s =$	0	1	2	3	4
$t = 2$	1	4	—	—	$t = 2$	1	4	—	—	—
$t = 3$	1	4	—	—	$t = 3$	1	4	—	—	—
$t = 4$	1	4	8	—	$t = 4$	1	4	8	—	—
$t = 5$	1	2	8	—	$t = 5$	1	4	8	—	—
$t = 6$	1	<b>2</b>	<b>2</b>	10	$t = 6$	1	4	8	11	—
					$t = 7$	1	4	8	10	—
					$t = 8$	1	4	8	9	10
					$t = 9$	1	4	<b>8</b>	<b>8</b>	10

the affine space  $\mathcal{K}_t^{\mathbb{C}}$ , thus replacing the semidefinite feasibility problem by a linear algebra computation. We refer to [18] for details on correctness and termination of this algorithm.

Alternatively one can describe the above situation as follows: the complex analogue of Algorithm 1 is an algorithm for complex roots, which can be turned into an algorithm for real roots simply by adding the positivity condition on  $\Lambda$ . This suggests that the same recipe could be applied to other algorithms for complex roots. This is indeed the case, for instance, for the prolongation-projection algorithm of [34] which, as shown in [19], can be turned into an algorithm for real roots by adding a positivity condition. The algorithm of [34] works with the space  $\mathcal{K}_t^{\mathbb{C}}$  but uses a different stopping criterion instead of the rank condition (2.14). Namely one should check whether, for some  $D \leq s \leq t$ , the three affine spaces  $\pi_s(\mathcal{K}_t^{\mathbb{C}})$ ,  $\pi_{s-1}(\mathcal{K}_t^{\mathbb{C}})$ , and  $\pi_s(\mathcal{K}_{t+1}^{\mathbb{C}})$  have the same dimensions (where  $\pi_s(\Lambda)$  denotes the restriction of  $\Lambda \in (\mathbb{R}[x])_t^*$  to  $(\mathbb{R}[x]_s)^*$ ); if so, one can compute a basis of  $\mathbb{R}[x]/I$  and extract  $V_{\mathbb{C}}(I)$ . Roughly speaking, to turn this into an algorithm for real roots, one adds positivity and considers the convex set  $\mathcal{K}_t$  instead of  $\mathcal{K}_t^{\mathbb{C}}$ ; again one needs to check that three suitably defined spaces have the same dimensions; if so, then one can extract  $V_{\mathbb{R}}(I)$ . We refer to [19] for details, also about the links between the rank condition and the above alternative stopping criterion.

*Example 2.10.* We apply the real vs. complex moment matrix algorithms to the ideal  $I = \langle h_1, h_2, h_3 \rangle$  (taken from [5, Example 4, p. 57]), where

$$h_1 = x_1^2 - 2x_1x_3 + 5, \quad h_2 = x_1x_2^2 + x_2x_3 + 1, \quad h_3 = 3x_2^2 - 8x_1x_3,$$

with  $D = 3$ ,  $|V_{\mathbb{C}}(I)| = 8$  and  $|V_{\mathbb{R}}(I)| = 2$ . Table 2.2 shows the ranks of the generic moment matrices when applying the real vs. complex versions of the moment matrix algorithm. We see that the algorithm terminates earlier in the real case, namely at order  $t = 6$ , compared to order  $t = 9$  in the complex case. If we replace each polynomial  $h_i$  by  $h_i \cdot (1 + \sum_i x_i^2)$ , we obtain an example with a positive dimensional complex variety, while the real variety is unchanged. The real root finding algorithm still terminates (now at order  $t = 7$ ) and allows the extraction of the two real roots.

## 2.4 Further Directions and Connections

The moment approach for real solving polynomial equations can be extended and applied in various directions. We briefly mentioned at the end of Sect. 2.3.2 the link to the approach of the Chap. 5 of this Handbook for approximating the convex hull of a real variety. We now touch a few selected extensions: polynomial optimization, emptiness certificates for real varieties, the positive dimensional case, and quotient ideals.

### 2.4.1 Optimization and Polynomial Inequalities

The research field of polynomial optimization, which roots, in particular, in work of Lasserre [15], Parrilo [33], Shor [38], has recently undergone a spectacular development. We refer e.g. to the monograph [16] or the survey [21] for overview and further references. The moment approach was originally proposed in [15] for solving general nonlinear optimization problems of the form

$$\begin{aligned} f^* = \min_x f(x) \text{ such that } h_1(x) = 0, \dots, h_m(x) = 0, \\ g_1(x) \geq 0, \dots, g_p(x) \geq 0, \end{aligned} \quad (2.19)$$

where  $f, h_i, g_j \in \mathbb{R}[x]$ . Let  $I = \langle h_1, \dots, h_m \rangle$  be the ideal generated by the  $h_i$ 's, and set

$$S = \{x \in \mathbb{R}^n \mid g_1(x) \geq 0, \dots, g_p(x) \geq 0\}, \quad (2.20)$$

so that (2.19) asks to minimize  $f$  over the semi-algebraic set  $V_{\mathbb{R}}(I) \cap S$ . The basic observation in [15] is that the problem (2.19) can be reformulated as

$$\min_{\mu} \Lambda_{\mu}(f) \text{ such that } \mu \text{ is a probability measure on } V_{\mathbb{R}}(I) \cap S,$$

where  $\Lambda_{\mu}$  is as in (2.9). Such a linear form satisfies:  $\Lambda(h) = 0$  for all  $h \in I$ , as well as the positivity condition:  $\Lambda(g_j f^2) \geq 0$  for all  $f \in \mathbb{R}[x]$  and  $j = 1, \dots, p$ . The latter conditions can be reformulated as requiring that the *localizing moment matrices*  $M_{\lfloor \frac{t-\deg(g_j)}{2} \rfloor}(g_j \Lambda)$  be positive semidefinite. Here, for  $g \in \mathbb{R}[x]$ ,  $g \Lambda$  is the new linear form defined by  $g \Lambda(p) = \Lambda(pg)$  for all  $p \in \mathbb{R}[x]$ .

The semidefinite program (2.16) can be modified in the following way to yield a relaxation of (2.19):

$$\begin{aligned} f_t^* = \min_{\Lambda \in (\mathbb{R}[x])^*} \Lambda(f) \text{ such that } \Lambda(1) = 1, \Lambda(h) = 0 \forall h \in \mathcal{H}_t, \\ M_{\lfloor \frac{t-\deg(g_j)}{2} \rfloor}(g_j \Lambda) \succeq 0 \ (j = 0, 1, \dots, p) \end{aligned} \quad (2.21)$$

(setting  $g_0 = 1$ ). The dual semidefinite program reads:

$$\max \lambda \text{ such that } f - \lambda = \sum_{j=0}^p \sigma_j g_j + \sum_{i=1}^m u_i h_i \quad (2.22)$$

where  $u_i \in \mathbb{R}[x]$ ,  $\sigma_j$  are sums of squares of polynomials with  $\deg(u_i h_i), \deg(\sigma_j g_j) \leq t$ . Then,  $f_t^* \leq f^*$  for all  $t$ . Moreover, asymptotic convergence of (2.21) and (2.22) to the minimum  $f^*$  of (2.19) can be shown when the feasible region of (2.19) is compact and satisfies some additional technical condition (see [15]). We now group some results showing finite convergence under certain rank condition, which can be seen as extensions of Theorems 2.11 and 2.12.

**Theorem 2.13 ([12, 17, 21]).** *Let  $D := \max_{i,j}(\deg(h_i), \deg(g_j))$ ,  $d := \lceil D/2 \rceil$ ,  $t \geq \max(\deg(f), D)$ , and let  $\Lambda$  be a generic optimal solution to (2.21) (i.e., for which  $\text{rank } M_{\lfloor t/2 \rfloor}(\Lambda)$  is maximum), provided it exists.*

- (i) *If the rank condition (2.15) holds with  $2s \geq \deg(f)$ , then  $f_t^* = f^*$  and  $V_{\mathbb{C}}(\text{Ker } M_s(\Lambda))$  is equal to the set of global minimizers of the program (2.19).*
- (ii) *If  $V_{\mathbb{R}}(I)$  is nonempty and finite, then (2.15) holds with  $2s \geq \deg(f)$ .*
- (iii) *If  $V_{\mathbb{R}}(I) = \emptyset$ , then the program (2.21) is infeasible for  $t$  large enough.*

In other words, under the rank condition (2.15), one can compute all global minimizers of the program (2.19), since, as before, one can compute a basis of the space  $\mathbb{R}[x]/\langle \text{Ker } M_s(\Lambda) \rangle$  from the moment matrix and thus apply the eigenvalue method. Moreover, when the equations  $h_i = 0$  have finitely many real roots, the rank condition is guaranteed to hold after finitely many steps.

By choosing the constant objective function  $f = 1$  in (2.19), we can also compute the  $S$ -radical ideal:

$$\sqrt[S]{I} := I(V_{\mathbb{R}}(I) \cap S).$$

When  $|V_{\mathbb{R}}(I)|$  is nonempty and finite, one can show that

$$I(V_{\mathbb{R}}(I) \cap S) = \langle \text{Ker } M_s(\Lambda) \rangle$$

for a generic optimal solution  $\Lambda$  of (2.21) and  $s, t$  large enough. An analogous result holds under the weaker assumption that  $|V_{\mathbb{R}}(I) \cap S|$  is nonempty and finite. In this case  $\Lambda$  needs to be a generic feasible solution of the modified semidefinite program obtained by adding to (2.21) the positivity conditions:

$$M_{\lfloor \frac{t-\deg(g)}{2} \rfloor}(g\Lambda) \succeq 0 \text{ for } g = g_1^{e_1} \cdots g_p^{e_p} \forall e \in \{0, 1\}^p.$$

The key ingredient in the proof is to use the Positivstellensatz to characterize the polynomials in  $I(V_{\mathbb{R}}(I) \cap S)$  (see [41]) instead of the Real Nullstellensatz (used in Theorem 2.10 to characterize the polynomials in  $I(V_{\mathbb{R}}(I))$ ).

Let us illustrate on an example how to ‘zoom in’ on selected roots, by incorporating semi-algebraic constraints or suitably selecting the cost function.

**Table 2.3** Ranks of  $M_s(\Lambda)$  in Example 2.11

(a) Generic $\Lambda \in \mathcal{K}_t$					(b) Generic $\Lambda \in \mathcal{K}_t$ with $M_{\lfloor \frac{t-1}{2} \rfloor}(g\Lambda) \succeq 0$			
$s =$	0	1	2	3	$s =$	0	1	2
$t = 2$	1	6	–	–	$t = 2$	1	6	–
$t = 3$	1	6	–	–	$t = 3$	1	6	–
$t = 4$	1	6	16	–	$t = 4$	1	<b>5</b>	<b>5</b>
$t = 5$	1	6	16	–				
$t = 6$	1	6	<b>12</b>	<b>12</b>				

*Example 2.11.* Consider the following system, known as Katsura 5 (see [13]):

$$\begin{aligned} h_1 &= 2x_6^2 + 2x_5^2 + 2x_4^2 + 2x_3^2 + 2x_2^2 + x_1^2 - x_1, \\ h_2 &= x_6x_5 + x_5x_4 + 2x_4x_3 + 2x_3x_2 + 2x_2x_1 - x_2, \\ h_3 &= 2x_6x_4 + 2x_5x_3 + 2x_4x_2 + x_2^2 + 2x_3x_1 - x_3, \\ h_4 &= 2x_6x_3 + 2x_5x_2 + 2x_3x_2 + 2x_4x_1 - x_4, \\ h_5 &= x_3^2 + 2x_6x_1 + 2x_5x_1 + 2x_4x_1 - x_5, \\ h_6 &= 2x_6 + 2x_5 + 2x_4 + 2x_3 + 2x_2 + x_1 - 1, \end{aligned}$$

with  $D = 2$ ,  $|V_{\mathbb{C}}(I)| = 32$ , and  $|V_{\mathbb{R}}(I)| = 12$ . Table 2.3a shows the ranks of the generic moment matrices for the moment matrix algorithm to compute  $V_{\mathbb{R}}(I)$ . At order  $(t, s) = (6, 3)$ , the algorithm finds all twelve real roots.

Next we apply the moment matrix algorithm to compute the real roots in  $S = \{x \in \mathbb{R}^6 \mid g(x) = x_1 - 0.5 \geq 0\}$ ; the ranks are shown in Table 2.3b and all five elements of  $V_{\mathbb{R}}(I) \cap S$  can be computed at order  $(t, s) = (4, 2)$ .

If we are interested e.g. only in the roots in  $V_{\mathbb{R}}(I) \cap S$  with the smallest  $x_2$ -coordinate then we minimize the polynomial  $x_2$  (instead of the constant one polynomial). The moment matrix algorithm now terminates at order  $(t, s) = (2, 1)$  and finds the unique element of  $V_{\mathbb{R}}(I) \cap S$  with the smallest  $x_2$ -coordinate.

#### 2.4.2 Exact Certificates of Emptiness

If the moment method is applied to an empty real variety  $V_{\mathbb{R}}(I)$  (or subset  $V_{\mathbb{R}}(I) \cap S$ ), then the underlying semidefinite optimization problem is infeasible for  $t$  large enough, which thus can be thought of as a *numerical certificate* of emptiness (see Theorems 2.12, 2.13). If we solve the semidefinite program (2.16) with a primal-dual interior point solver and infeasibility is detected, an improving ray is returned, i.e., a solution to the dual problem (2.17) of the form:

$$1 - \lambda^* = \sigma + \sum_{i=1}^m u_i h_i \quad \text{where } \sigma, u_i \in \mathbb{R}[x] \text{ and } \sigma \text{ is a sum of squares,} \quad (2.23)$$

with  $\lambda^* > 1$ . By scaling both sides with an arbitrary positive number, one can generate a feasible solution of the dual problem (2.17) with an arbitrary high cost function value, thus certifying infeasibility of the primal problem.

On the other hand, by the Real Nullstellensatz, we know that an algebraic certificate for emptiness of  $V_{\mathbb{R}}(I)$  is that  $1 \in \sqrt[{\mathbb{R}}]{I}$ , i.e.,

$$1 + \sigma = \sum_{i=1}^m u_i h_i \text{ for some } \sigma, u_i \in \mathbb{R}[x] \text{ where } \sigma \text{ is a sum of squares.} \quad (2.24)$$

In principle, such a certificate can be directly derived from an improving ray such as (2.23). The difficulty, however, arise from numerical imprecisions and the certificate computed using semidefinite programming does not hold exactly when all computations are done in floating point arithmetics. We may thus only derive polynomials  $u_i, \sigma$  satisfying

$$1 + \sigma + \epsilon = \sum_{i=1}^m u_i h_i, \quad (2.25)$$

where  $\epsilon \in \mathbb{R}[x]$ , represents the cumulated error term. However, as shown in [35, Proposition 7.38], this approximate certificate can still be used to produce an *exact* certificate for the nonexistence of roots in some ball  $B_\delta$  of radius  $\delta$  around the origin. Namely, if  $|\epsilon(0)| \ll 1$ , then one can compute an explicit  $\delta$  for which one can prove that  $V_{\mathbb{R}}(I) \cap B_\delta = \emptyset$ . This is illustrated on the following example.

*Example 2.12.* Consider the ideal  $I = \langle h_1, h_2, h_3 \rangle$  generated by

$$h_1 = x_1^4 + x_2^4 + x_3^4 - 4, \quad h_2 = x_1^5 + x_2^5 + x_3^5 - 5, \quad h_3 = x_1^6 + x_2^6 + x_3^6 - 6$$

with  $D = 6$ ,  $|V_{\mathbb{C}}(I)| = 120$ , and  $V_{\mathbb{R}}(I) = \emptyset$ . At order  $t = 6$  already, the primal (moment) problem is infeasible, the solver returns an improving direction for the dual (SOS) problem, and we obtain a numerical certificate of the form (2.25). The error polynomial  $\epsilon \in \mathbb{R}[x]$  is a dense polynomial of degree 6, its coefficients are smaller than  $4.1e-11$ , with constant term  $\epsilon(0) < 8.53e-14$ . Using the conservative estimate of [35, Sect. 7.8.2] one can rigorously certify the emptiness of the set  $V_{\mathbb{R}}(I) \cap B_\delta$  for  $\delta = 38.8$ . In other words, even if we only solved the problem numerically with a rather low accuracy, we still obtain a proof that the ideal  $I$  does not have any real root  $v \in V_{\mathbb{R}}(I)$  with  $\|v\|_2 < 38.8$ . By increasing the accuracy of the SDP solver the radius  $\delta$  of the ball can be further increased. This example illustrates that it is sometimes possible to draw exact conclusions from numerical computations.

### 2.4.3 Positive Dimensional Ideals and Quotient Ideals

Dealing with positive dimensional varieties is a challenging open problem, already for complex varieties (see e.g. the discussion in [24]). The algorithm presented so far for computing the real variety  $V_{\mathbb{R}}(I)$  and the real radical ideal  $\sqrt[{\mathbb{R}}]{I}$  works under the assumption that  $V_{\mathbb{R}}(I)$  is finite. Indeed, the rank condition (2.14) (or (2.15))

implies that  $\dim \mathbb{R}[x]/\sqrt[\mathbb{R}]{I} = \text{rank } M_{s-1}(\Lambda)$  is finite (by Theorem 2.11). Nevertheless, the moment method can in principle be applied to find a basis of  $\sqrt[\mathbb{R}]{I}$  also in the positive dimensional case. Indeed Theorem 2.10 shows that, for  $t$  large enough, the kernel of  $M_{\lfloor t/2 \rfloor}(\Lambda)$  (for generic  $\Lambda \in \mathcal{K}_t$ ) generates the real radical ideal  $\sqrt[\mathbb{R}]{I}$ . The difficulty however is that it is not clear how to recognize whether equality  $\sqrt[\mathbb{R}]{I} = \langle \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda) \rangle$  holds in the positive dimensional case. These questions relate in particular to the study of the Hilbert function of  $\sqrt[\mathbb{R}]{I}$  (see [35]). An interesting research direction is whether the moment matrix approach can be applied to compute some form of “parametric representation” of the real variety. On some instances it is indeed possible to compute parametric multiplication matrices (see [35] for details).

Another interesting object is the quotient (or colon) ideal

$$I : g = \{p \in \mathbb{R}[x] \mid pg \in I\}$$

for an ideal  $I$  and  $g \in \mathbb{R}[x]$ . The moment approach can be easily adapted to find a semidefinite characterization of the ideal  $\sqrt[\mathbb{R}]{I} : g = I(V_{\mathbb{R}}(I) \setminus V_{\mathbb{R}}(g))$ . Indeed, for generic  $\Lambda$ , the kernel of the localizing moment matrix of  $g\Lambda$  carries all information about this ideal.

**Proposition 2.1.** *Let  $g \in \mathbb{R}[x]_k$ ,  $\rho := 1 + \lceil k/2 \rceil$  and  $D = \max_i \deg(h_i)$ . Let  $\Lambda$  be a generic element in  $\mathcal{K}_{t+k}$ .*

- (i)  $\langle \text{Ker } M_{\lfloor t/2 \rfloor}(g\Lambda) \rangle \subseteq \sqrt[\mathbb{R}]{I} : g$ , with equality for  $t$  large enough.
- (ii) If the rank condition:  $\text{rank } M_s(\Lambda) = \text{rank } M_{s-\rho}(\Lambda)$  holds for some  $s$  with  $\max(D, \rho) \leq s \leq \lfloor t/2 \rfloor$ , then  $\sqrt[\mathbb{R}]{I} : g = \langle \text{Ker } M_{s-\rho+1}(g\Lambda) \rangle$ .
- (iii) If  $V_{\mathbb{R}}(I)$  is nonempty finite, then the rank condition in (ii) holds at some order  $(t, s)$ .

*Proof.* Note that  $p \in \text{Ker } M_{\lfloor t/2 \rfloor}(g\Lambda)$  if and only if  $pg \in \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$  when  $\deg(p) \leq \lfloor t/2 \rfloor - k$ .

- (i) As  $\Lambda$  is generic,  $\text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda) \subseteq \sqrt[\mathbb{R}]{I}$ , implying  $\langle \text{Ker } M_{\lfloor t/2 \rfloor}(g\Lambda) \rangle \subseteq \sqrt[\mathbb{R}]{I} : g$ . The proof of equality for  $t$  large enough is similar to the proof of Theorem 2.10: Pick a basis  $\{g_1, \dots, g_L\}$  of the ideal  $\sqrt[\mathbb{R}]{I} : g$ , so that each  $g_l g$  belongs to  $\sqrt[\mathbb{R}]{I}$ ; apply the Real Nullstellensatz to  $g_l g$  to conclude that, for  $t$  large enough,  $g_l g \in \text{Ker } M_{\lfloor t/2 \rfloor}(\Lambda)$  and thus  $g_l \in \text{Ker } M_{\lfloor t/2 \rfloor}(g\Lambda)$ .
- (ii) Assume now  $\text{rank } M_s(\Lambda) = \text{rank } M_{s-\rho}(\Lambda)$  for  $D, \rho \leq s \leq \lfloor t/2 \rfloor$ . Then there exists  $\tilde{\Lambda} \in \mathbb{R}[x]^*$  for which  $M(\tilde{\Lambda})$  is a flat extension of  $M_s(\tilde{\Lambda})$  and  $\sqrt[\mathbb{R}]{I} = \text{Ker } M(\tilde{\Lambda}) = \langle \text{Ker } M_{s-\rho+1}(\Lambda) \rangle$  (use Theorems 2.9 and 2.11). Therefore,  $\sqrt[\mathbb{R}]{I} : g = \text{Ker } M(\tilde{\Lambda}) : g = \text{Ker } M(g\tilde{\Lambda})$ . One can verify that  $M(g\tilde{\Lambda})$  is a flat extension of  $M_{s-\rho}(g\tilde{\Lambda})$ , which implies that  $\text{Ker } M(g\tilde{\Lambda}) = \langle \text{Ker } M_{s-\rho+1}(g\tilde{\Lambda}) \rangle$  (using Theorem 2.9) is thus equal to  $\langle \text{Ker } M_{s-\rho+1}(g\Lambda) \rangle$  (since  $\tilde{\Lambda}$  and  $\Lambda$  coincide on  $\mathbb{R}[x]_{2s}$ ).
- (iii) Follows from an easy modification of the proof of Theorem 2.12.  $\square$

We conclude this chapter with a small example on quotient ideals.

**Table 2.4** Rank sequences for generic  $\Lambda \in \mathcal{K}_t$  in Example 2.13

(a) rank $M_s(\Lambda)$					(b) rank $M_s(g\Lambda)$				
$s =$	0	1	2	3	$s =$	0	1	2	3
$t = 4$	1	3	3	–	$t = 4$	1	2	–	–
$t = 5$	1	3	3	–	$t = 5$	1	2	–	–
$t = 6$	1	<b>3</b>	3	<b>3</b>	$t = 6$	1	<b>2</b>	<b>2</b>	–

*Example 2.13.* Consider  $I = \langle x_2 - x_1^2, x_2^2 - x_2 \rangle$ , with roots  $(0,0)$  (double) and  $(\pm 1, 1)$ , and  $\sqrt[3]{I} = \langle x_2 - x_1^2, x_2^2 - x_2, x_1 x_2 - x_1 \rangle$ . The quotient ideal computation with  $g = x_1$  terminates at order  $(t, s) = (6, 3)$  and we obtain that  $\sqrt[3]{I} : g$  is generated by  $x_2 - x_1^2, x_2 - 1$ , with variety  $V_{\mathbb{R}}(I) \setminus V_{\mathbb{R}}(g) = \{(-1, 1), (1, 1)\}$ . The corresponding ranks of  $M_s(\Lambda)$  and  $M_s(g\Lambda)$  are shown in Table 2.4.

## References

1. Akhiezer, N.I.: The Classical Moment Problem, Hafner (1965)
2. Basu, S., Pollack, R., Roy, M.-F.: Algorithms in Real Algebraic Geometry, Springer (2003)
3. Bates, D., Sottile, F.: Khovanskii-Rolle continuation for real solutions, arXiv:0908.4579 (2009)
4. Bochnak, J., Coste, M., Roy, M.-F.: Real Algebraic Geometry, Springer (1998)
5. Cox, D.A., Little, J.B., O’Shea, D.B.: Using Algebraic Geometry, Springer (1998)
6. Cox, D.A., Little, J.B., O’Shea, D.B.: Ideals, Varieties, and Algorithms : An Introduction to Computational Algebraic Geometry and Commutative Algebra, Springer (2005)
7. Curto, R.E., Fialkow, L.: Solution of the truncated complex moment problem for flat data. *Memoirs of the American Mathematical Society* **119**(568), 1–62 (1996)
8. de Klerk, E.: Aspects of Semidefinite Programming – Interior Point Algorithms and Selected Applications, Kluwer (2002)
9. Dickenstein, A., Emiris, I.Z. (eds.): Solving Polynomial Equations: Foundations, Algorithms, and Applications. Algorithms and Computation in Mathematics, vol. 14, Springer (2005)
10. Elkadi, M., Mourrain, B.: Introduction à la Résolution des Systèmes d’Équations Polynomiaux. Mathématiques et Applications, vol. 59, Springer (2007)
11. Haviland, E.K.: On the momentum problem for distribution functions in more than one dimension. II. *American Journal of Mathematics* **58**(1), 164–168 (1936)
12. Henrion, D., Lasserre, J.B.: Detecting global optimality and extracting solutions in GloptiPoly. In: Positive Polynomials in Control, Lecture Notes in Control and Information Sciences, pp. 293–310, Springer (2005)
13. <http://www.mat.univie.ac.at/~neum/glopt/coconut/Benchmark/Library3/\katsura5.mod>
14. Kehrein, A., Kreuzer, M.: Characterizations of border bases. *Journal of Pure and Applied Algebra* **196**, 251–270 (2005)
15. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization* **11**, 796–817 (2001)
16. Lasserre, J.B.: Moments, Positive Polynomials, and their Applications, Imperial College Press (2009)
17. Lasserre, J.B., Laurent, M., Rostalski, P.: Semidefinite characterization and computation of real radical ideals. *Foundations of Computational Mathematics* **8**(5), 607–647 (2008)
18. Lasserre, J.B., Laurent, M., Rostalski, P.: A unified approach for real and complex zeros of zero-dimensional ideals.. In: Putinar, M., Sullivant, S. (eds.) Emerging Applications of Algebraic Geometry, vol. 149, pp. 125–156. Springer (2009)

19. Lasserre, J.B., Laurent, M., Rostalski, P.: A prolongation-projection algorithm for computing the finite real variety of an ideal. *Theoretical Computer Science* **410**(27–29), 2685–2700 (2009)
20. Laurent, M.: Revisiting two theorems of Curto and Fialkow. *Proceedings of the American Mathematical Society* **133**(10), 2965–2976 (2005)
21. Laurent, M.: Sums of squares, moment matrices and optimization over polynomials. In: Putinar, M., Sullivant, S. (eds.) *Emerging Applications of Algebraic Geometry*, pp. 157–270. Springer (2009)
22. Laurent, M., Mourrain, B.: A generalized flat extension theorem for moment matrices. *Archiv der Mathematik* **93**(1), 87–98 (2009)
23. Lazard, D.: Resolution des systèmes d'équations algébriques. *Theoretical Computer Science* **15**, 77–110 (1981)
24. Lazard, D.: Thirty years of polynomial system solving, and now? *Journal of Symbolic Computation* **44**, 222–231 (2009)
25. Löfberg, J.: YALMIP: A Toolbox for Modeling and Optimization in MATLAB. *Computer Aided Control Systems Design Conference* (Taipei, Taiwan). Available from <http://users.isy.liu.se/johanl/yalmip/> (2004)
26. Möller, H.M.: An inverse problem for cubature formulae. *Computational Technologies* **9**, 13–20 (2004)
27. Mora, T.: Solving Polynomial Equation Systems I: The Kronecker-Duval Philosophy. *Encyclopedia of Mathematics and its Applications*, vol. 88. Cambridge University Press (2003)
28. Mora, T.: Solving Polynomial Equation Systems II: Macaulay's Paradigm and Gröbner Technology. *Encyclopedia of Mathematics and its Applications* (v. 2), vol. 99. Cambridge University Press (2005)
29. Mourrain, B.: A new criterion for normal form algorithms. In: Fosserier, M., Imai, H., Lin, S., Poli, A. (eds.) *13th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, Honolulu, November 1999. *Lecture Notes in Computer Science*, vol. 1719, pp. 430–443. Springer (1999)
30. Mourrain, B.: Pythagore's dilemma, symbolic - numeric computation and the border basis method. In: *Symbolic-numeric computation, Trends in Mathematics*, pp. 223–243. Birkhäuser (2007)
31. Mourrain, B., Pavone, J.P.: Subdivision methods for solving polynomial equations. *Journal of Symbolic Computation* **44**(3), 292–306 (2009)
32. Mourrain, B., Trebuchet, P.: Generalized normal forms and polynomials system solving. In: Kauers, M. (ed.) *Proc. Intern. Symp. on Symbolic and Algebraic Computation*, pp. 253–260. ACM Press (2005)
33. Parrilo, P.A.: Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization. Ph.D. thesis, California Institute of Technology (2000)
34. Reid, G., Zhi, L.: Solving polynomial systems via symbolic-numeric reduction to geometric involutive form. *Journal of Symbolic Computation* **44**(3), 280–291 (2009)
35. Rostalski, P.: Algebraic Moments – Real Root Finding and Related Topics, Ph.D. thesis, ETH Zürich (2009)
36. Rostalski, P.: Bermeja: Convex Algebraic Geometry in MATLAB, Available from <http://math.berkeley.edu/~philipp/cagwiki>.
37. Rouillier, F.: Solving zero-dimensional systems through the rational univariate representation. *Journal of Applicable Algebra in Engineering, Communication and Computing* **9**(5), 433–461 (1999)
38. Shor, N.Z.: An approach to obtaining global extrema in polynomial mathematical programming problems. *Kibernetika* **5**, 102–106 (1987)
39. Sommese, A.J., Wampler, C.W.: *The Numerical Solution of Systems of Polynomials Arising in Engineering and Science*, World Scientific Press (2005)
40. Ottlieb, F.: *Real Solutions to Equations from Geometry*. University Lecture Series, vol. 57, AMS, (2011)
41. Stengle, G.: A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. *Mathematische Annalen* **207**, 87–97 (1974)

42. Stetter, H.J.: Numerical Polynomial Algebra, SIAM (2004)
43. Sturmfels, B.: Solving Systems of Polynomial Equations, American Mathematical Society (2002)
44. Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.): Handbook of Semidefinite Programming: Theory, Algorithms, and Applications, Vol. 27. Kluwer Academic Publishers, Dordrecht, The Netherlands / Boston, MA (2000)

# Chapter 3

## Algebraic Degree in Semidefinite and Polynomial Optimization

Kristian Ranestad

### 3.1 Introduction

In semidefinite and polynomial optimization problems both the constraint and the objective functions are defined by multivariate polynomials. When an optimizer exists, it has to satisfy some necessary optimal conditions that are algebraic. The coordinates of optimizers are therefore algebraic functions of the coefficients of these polynomials. For each coordinate there is a minimal univariate polynomial defined over the field generated by these coefficients. In general the degree of this polynomial is independent of the coordinate, and is called the *algebraic degree* of the optimization problem. Elimination theory (cf. [1, Chap. 3]) may be used to find this polynomial, but there is no general algorithm to find its roots. The algebraic degree is therefore a measure of complexity which is different in spirit to the standard computational complexity from computer science, and as such, should be viewed as a complement to the latter.

We shall give a number of formulas for the algebraic degree for various polynomial optimization problems, such as semidefinite programming (SDP), quadratically constrained quadratic programming (QCQP) and in the general polynomial case (POP). For complete proofs see [5, 6] and [4].

Although some terminology from algebraic geometry is needed, we shall keep it to a minimum and introduce the key notions when they naturally appear. The textbook [3] is recommended for further details.

---

K. Ranestad (✉)

Department of Mathematics, University of Oslo, PO Box 1053 Blindern,  
NO-0316 Oslo, Norway

e-mail: [ranestad@math.uio.no](mailto:ranestad@math.uio.no)

### 3.2 The Optimization Problem

Consider the problem

$$\begin{cases} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) = 0, i = 1, \dots, m_e \\ & \text{and } f_i(x) \geq 0, i = m_e + 1, \dots, m \end{cases} \quad (3.1)$$

where the  $f_0(x), \dots, f_m(x)$  are multivariate polynomial functions in  $\mathbb{R}[x]$ , the ring of polynomials in  $x = (x_1, \dots, x_n)$ .

If  $f_* = f_0(p)$  is a solution to this problem and all the inequality constraints at  $p$  are active, i.e.  $f_i(p) = 0$  for  $i = 1, \dots, m$ , then  $p \in \mathbb{R}^n$  is a critical point for the polynomial  $f_0$  restricted to the set

$$K = K(f_1, \dots, f_m) = \{x \in \mathbb{R}^n | f_i(x) = 0, i = 1, \dots, m\},$$

defined by the constraint functions. If some of the constraint functions are inactive, i.e.  $f_i(p) > 0$  for some  $i > m_e$ , then we may proceed defining  $K$  with the active constraints. The set  $K$ , being the common zero locus of a collection of multivariate polynomials, is an *affine variety*. The set  $K$  may be smooth, i.e. a manifold, or it may be singular. Also,  $K$  may have several irreducible components of different dimensions. With additional constraints we may separate components of different dimensions, so we may assume that all components of  $K$  have the same dimension. If the codimension of  $K$  is  $k$  and  $p$  is a smooth point on  $K$ , then the gradient vectors of the polynomials  $f_i$  at  $p$  generate a  $k$ -dimensional vector space, orthogonal to the tangent space to  $K$  at  $p$ . The variety  $K$  is singular at the points where these gradient vectors fail to span a  $k$ -dimensional vector space. Therefore the singular locus  $\text{sing } K$  is the affine subvariety of  $K$  defined by the  $k \times k$  minors of the Jacobian matrix

$$\begin{pmatrix} \frac{\partial}{\partial x_1} f_1(x) & \frac{\partial}{\partial x_1} f_2(x) & \dots & \frac{\partial}{\partial x_1} f_m(x) \\ \frac{\partial}{\partial x_2} f_1(x) & \frac{\partial}{\partial x_2} f_2(x) & \dots & \frac{\partial}{\partial x_2} f_m(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_n} f_1(x) & \frac{\partial}{\partial x_n} f_2(x) & \dots & \frac{\partial}{\partial x_n} f_m(x) \end{pmatrix}.$$

In particular  $\text{sing } K$  is again an affine variety. Its complement  $K_0 = K \setminus \text{sing } K$  is the open set of nonsingular points on  $K$ . As an affine variety,  $\text{sing } K$  has itself a singular subvariety. Let  $K_1$  be the complement of the singular locus in  $\text{sing } K$ . This process may be iterated until we get a finite set of components:  $K = K_0 \cup K_1 \cup \dots \cup K_r$ , such that each  $K_i$  is smooth, and the closure of each  $K_{i+1}$  is defined by the appropriate minors of a Jacobian matrix as the singular locus in the closure of  $K_i$ . This decomposition of  $K$  reduces the original optimization problem to the case where  $K$  is smooth, i.e. has no singular points.

The objective function  $f_0$  defines level sets  $L_a = \{x \in \mathbb{R}^n | f_0(x) = a\}$ , and the critical locus of  $f_0$  on  $K$  is the subset  $C(f_0) \subset K$  of points on  $K$ , such that the tangent hyperplane to  $L_{f_0(p)}$  contains the tangent plane to  $K$  at  $p$ . With Lagrangian multipliers; the gradient  $\nabla f_0(p) = \sum_{i=1}^m \lambda_i \nabla f_i(p)$  for suitable  $\lambda_i$ .

It is a fundamental fact that, given a smooth algebraic variety  $K \subset \mathbb{R}^n$ , the critical locus is a finite set for the general objective function  $f_0$ , i.e. for any objective function in an open set in the set of functions.

The algebraic degree  $\delta$  is then simply the cardinality of the critical locus  $C(f_0)$ .

In the examples below we shall always assume that the initial data are general. Unless otherwise stated, we shall assume that  $K$  is smooth. In the case of semidefinite programming, however,  $K$  is singular. In this case we shall decompose  $K$  into smooth strata  $K_i$  as above, and for general initial data compute the algebraic degree  $\delta_i$  as the cardinality of the critical locus of the objective function on each stratum  $K_i$ .

### 3.3 The Derivation of Algebraic Degree

#### 3.3.1 The Complex Projective Setup

In the previous section we saw that the algebraic degree for a polynomial optimization problem is a number of common solutions of a set of polynomial equations. The number of solutions of a general polynomial equation in one variable equals the degree of the polynomial, if you count all the complex solutions. This may be generalized to sets of multivariate polynomial equations. Therefore we extend our scalars to the complex numbers. Also, it is natural to allow solutions of the polynomial equations at infinity. Put together, this means that we extend our affine space  $\mathbb{R}^n$  to complex projective space  $\mathbb{CP}^n$ . This space is defined as the set of one dimensional subspaces of the vector space  $\mathbb{C}^{n+1}$ . Each point in  $\mathbb{CP}^n$  have homogeneous coordinates  $(p_0 : \dots : p_n)$  where the vector  $(p_0, \dots, p_n)$  is well defined up to nonzero scalar multiple. We denote the corresponding homogeneous coordinate functions by  $X_0, \dots, X_n$ . The map

$$\mathbb{C}^n \rightarrow \mathbb{CP}^n : (p_1, \dots, p_n) \mapsto (1 : p_1 : \dots : p_n)$$

identifies the affine complex space  $\mathbb{C}^n$  as a subset of the projective space. Thus  $\mathbb{CP}^n$  can be interpreted as the projective space extending the affine complex space  $\mathbb{C}^n$  to the hyperplane  $\{X_0 = 0\}$  at infinity. The common zero locus in  $\mathbb{CP}^n$  of a collection of homogeneous polynomials in the homogeneous coordinate functions is called a *projective variety*. Any polynomial  $f(x)$  of degree  $d$  in  $x = (x_1, \dots, x_n)$  may be homogenized to a homogeneous polynomial  $F(X) = X_0^d f(\frac{X_1}{X_0}, \dots, \frac{X_n}{X_0})$ , a homogeneous polynomial of degree  $d$  in the coordinates  $X = (X_0, \dots, X_n)$ . Since  $f(p_1, \dots, p_n) = 0$

if and only if  $F(1 : p_1 : \dots : p_n) = 0$ , the zero locus of  $f$  in  $\mathbb{C}^n$  coincides with zero locus of  $F$  in  $\mathbb{CP}^n \setminus \{X_0 = 0\}$ . The projective variety

$$\bar{K} = \bar{K}(f_1, \dots, f_m) = \{x \in \mathbb{CP}^n \mid F_i(x) = 0, i = 1, \dots, m\},$$

in  $\mathbb{CP}^n$  therefore contains the affine variety  $K(f_1, \dots, f_m)$  as a subset. Projective varieties define the closed subsets of the Zariski topology on  $\mathbb{CP}^n$  ([3, pp. 17–18]), so  $\bar{K}$  is the Zariski closure of the subset  $K(f_1, \dots, f_m)$  in  $\mathbb{CP}^n$ .

Each level set  $L_a = \{x \in \mathbb{R}^n \mid f_0(x) = a\}$  of the objective function extends in  $\mathbb{CP}^n$  to a hypersurface defined by  $F_{0,a}(X) = F_0(X) - aX_0^{d_0}$  where  $d_0$  is the degree of  $f_0$ . The critical set  $C(f_0)$  on  $K$ , now extends to the critical set  $C(F_0)$  on  $\bar{K}$ . The homogeneous coordinates of a critical point  $p = (p_1, \dots, p_n) \in \mathbb{R}^n$  are  $\bar{p} = (1 : p_1 : \dots : p_n)$ . If  $f_0(p) = a$ , then  $F_{0,a}(\bar{p}) = 0$  and the extended Jacobian matrix

$$\begin{pmatrix} \frac{\partial}{\partial X_0} F_{0,a}(\bar{p}) & \frac{\partial}{\partial X_1} F_1(\bar{p}) & \dots & \frac{\partial}{\partial X_0} F_m(\bar{p}) \\ \frac{\partial}{\partial X_1} F_{0,a}(\bar{p}) & \frac{\partial}{\partial X_1} F_1(\bar{p}) & \dots & \frac{\partial}{\partial X_1} F_m(\bar{p}) \\ \frac{\partial}{\partial X_2} F_{0,a}(\bar{p}) & \frac{\partial}{\partial X_2} F_1(\bar{p}) & \dots & \frac{\partial}{\partial X_2} F_m(\bar{p}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial X_n} F_{0,a}(\bar{p}) & \frac{\partial}{\partial X_n} F_1(\bar{p}) & \dots & \frac{\partial}{\partial X_n} F_m(\bar{p}) \end{pmatrix}, \quad (3.2)$$

has rank at most  $\text{codim } K$ . We may eliminate  $a$ , if we assume that there are no critical points at infinity,  $\{X_0 = 0\}$ . In fact, when  $F_k(X) = 0$  and  $X_0 \neq 0$ , then, by the Euler relation,

$$X_0 \frac{\partial}{\partial X_0} F_k(X) = d_k F_k(X) - \sum_{j=1}^n \frac{\partial}{\partial X_j} F_k(X) = - \sum_{j=1}^n \frac{\partial}{\partial X_j} F_k(X),$$

where  $d_k$  is the degree of  $F_k$ . So the matrix (3.2) has rank at most  $\text{codim } K$  if and only if the reduced matrix

$$M(\bar{p}) = \begin{pmatrix} \frac{\partial}{\partial X_1} F_0(\bar{p}) & \frac{\partial}{\partial X_1} F_1(\bar{p}) & \dots & \frac{\partial}{\partial X_1} F_m(\bar{p}) \\ \frac{\partial}{\partial X_2} F_0(\bar{p}) & \frac{\partial}{\partial X_2} F_1(\bar{p}) & \dots & \frac{\partial}{\partial X_2} F_m(\bar{p}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial X_n} F_0(\bar{p}) & \frac{\partial}{\partial X_n} F_1(\bar{p}) & \dots & \frac{\partial}{\partial X_n} F_m(\bar{p}) \end{pmatrix} \quad (3.3)$$

has rank at most  $\text{codim}K$ . Let

$$C_{\bar{K}} = \{X \in \mathbb{CP}^n : \text{rk}M(X) \leq \text{codim}K\}.$$

Then both  $\bar{K}$  and  $C_{\bar{K}}$  are algebraic subsets of  $\mathbb{CP}^n$ , and their intersection is the set of critical points, i.e.

$$C(F_0) = C_{\bar{K}} \cap \bar{K}.$$

When this set is finite its cardinality is an upper bound for the algebraic degree. Under certain genericity conditions this upper bound is attained and the algebraic degree coincides with the cardinality of this critical set.

### 3.3.2 Genericity Assumptions and KKT

Each optimization problem has its own algebraic degree. A formula or an effective algorithm to compute this degree in every case is out of reach. Even the above approach makes sense only under some genericity assumptions. We summarize these here for reference and completeness.

#### 3.3.2.1 Necessary Conditions

The following conditions are necessary for the above computation of the degree of the critical locus  $\bar{K} \cap C_{\bar{K}}$  to make sense:

1.  $K$  is equidimensional, i.e. every irreducible component have the same dimension.
2. Each subset  $K_i$  is equidimensional.
3. The critical locus is finite, and contains no point at infinity.

The first two concerns the constraint functions, and may be achieved by possibly adding more constraints and consider one component at a time. The third condition depends on the objective function. Modifying this function or adding more constraints may reduce the critical set to a finite set.

#### Upper Bound

If the critical set is finite, its cardinality is an upper bound for the algebraic degree. This cardinality is the degree of a polynomial  $p(x)$  in one variable whose solutions are one coefficient of the critical points. The coefficients of the polynomials involved in computing the critical locus are rational functions in the coefficients of the original problem. These coefficients generate a subfield of  $\mathbb{R}$ , and a general polynomial with coefficients in this subfield is irreducible and have only simple roots in  $\mathbb{C}$ , in which case the upper bound is attained. If the polynomial  $p(x)$  is reducible, then the algebraic degree is necessarily smaller.

## KKT

The *Karush–Kuhn–Tucker conditions* [7, Chap. 10] are necessary local conditions for a minimum. In our notation they require, first, that the gradient of the objective function does not vanish at  $\bar{p}$ , secondly, that  $K$  has codimension equal to the number of active constraints and is nonsingular at  $\bar{p}$ , and, finally, for minimality, that the Lagrangian multipliers for the active inequality constraints are non-negative. Notice that the Lagrangian multipliers in this context are simply the coefficients expressing the linear dependence of the columns in  $M(\bar{p})$ .

### 3.3.3 Algebra Geometric Methods

Our computation of the cardinality of the critical locus will involve techniques and results from intersection theory. The intersection of an irreducible projective variety of codimension  $k$  in  $\mathbb{CP}^N$  with a general linear space of dimension  $k$  is a finite set of points. The cardinality of this intersection is called the *degree* of the variety. This notion of degree extends additively to reducible varieties whose components all have the same dimension. In particular, the degree of the variety defined by a homogeneous polynomial of degree  $d$  without multiple factors is again  $d$ . We shall compute the cardinality of the critical locus as its degree as a projective variety.

Intersection theory provides effective tools to compute the degree or cardinality of the intersection between complex projective varieties, but in general not to varieties in real affine space  $\mathbb{R}^n$  or real projective space. This reflects precisely the extension above of our varieties to complex projective space. The computations apply under some smoothness conditions, which are guaranteed by a basic result that follows from [3, Theorem 17.16].

**Theorem 3.1 (Bertini’s Theorem).** *Let  $Z \subset \mathbb{CP}^n$  be a projective variety, and  $G_0, \dots, G_r$  be homogeneous forms of the same degree on  $\mathbb{CP}^n$  that do not vanish on all of  $Z$ , then for a general choice of  $\alpha = [\alpha_0 : \dots : \alpha_r]$ , i.e. any point  $\alpha \in \mathbb{CP}^r$  outside some projective subvariety, the locus  $\{\sum_{i=0}^r \alpha_i G_i = 0\}$  intersects  $Z$  in a variety of codimension one on  $Z$  that is smooth away from  $\text{sing}(Z)$  and the locus  $\{G_0 = \dots = G_r = 0\}$ .*

In particular, this theorem says that the above definition of degree as the cardinality of the intersection of an irreducible variety of codimension  $k$  in  $\mathbb{CP}^N$  with a general linear space of dimension  $k$  makes sense.

To compute the degree in general we often use a result that follows from [3, Theorem 18.3].

**Theorem 3.2 (Bezout’s Theorem).** *Let  $Z$  and  $W$  be irreducible subvarieties of  $\mathbb{CP}^n$  of dimensions  $k$  and  $l$  with  $k + l \geq n$ . If the tangent spaces to  $Z$  and  $W$  at the general point of each irreducible component of the intersection  $Z \cap W$  intersect in dimension  $k + l - n$ , then*

$$\deg Z \cap W = \deg Z \cdot \deg W.$$

In particular, if  $k + l = n$ , then  $Z \cap W$  consists of  $\deg Z \cdot \deg W$  points.

The following situations will be of particular interest to us.

### Complete Intersections

If  $F_1, \dots, F_m$  are homogeneous polynomials in  $\mathbb{C}[X_0, \dots, X_n]$  that generate the ideal of the projective variety  $Z = \{x \in \mathbb{CP}^n | F_1(x) = \dots = F_m(x) = 0\}$  and  $Z$  has codimension  $m$  in  $\mathbb{CP}^n$ , then  $Z$  is called a complete intersection. For a general set of  $m$  polynomials, the variety  $Z$  is a complete intersection and, by Theorem 3.1, it is also nonsingular. Furthermore, if the polynomials  $F_1, \dots, F_m$  have degree  $d_1, \dots, d_m$ , respectively, then it follows from Theorem 3.2 that  $Z$  has degree

$$\deg Z = d_1 d_2 \cdots d_m.$$

### Determinantal Varieties

Let  $M = (m_{ij})$  be an  $(n \times m)$ -matrix with homogeneous polynomial entries  $m_{ij} \in \mathbb{C}[X_0, \dots, X_N]$ , and assume that there are positive integers  $a_1, \dots, a_m$  such that the degree of the polynomials  $m_{ij}$  in column  $j$  is  $a_j$ . Let  $Z_k$  be the projective variety defined by the  $(k+1) \times (k+1)$  minors of  $M$ . Thus  $Z_k$  is the locus of points where  $M$  has rank at most  $k$ . Since  $Z_k$  is defined by the minors of fixed size of a matrix it is called determinantal. The codimension of  $Z_k$  in the projective space  $\mathbb{P}^N$  is at most  $(n-k)(m-k)$ , by [3, Proposition 17.25]. The following proposition is obtained combining [3, Example 14.16] with Theorem 3.1.

**Proposition 3.1.** *Let  $M$  be an  $(n \times m)$ -matrix with general polynomial entries, then the subvariety  $Z_k$  where  $M$  has rank at most  $k$  has codimension equal to the minimum of  $(n-k)(m-k)$  and  $N+1$ , and is singular precisely along  $Z_{k-1}$ .*

The Giambelli–Thom–Porteous formula, [2, Theorem 14.4], computes the degree of  $Z_k$ , if it is nonempty and the matrix entries are general. In our notation it takes the form

$$\deg Z_k = \Delta_{n-k}^{(m-k)}(c) = \det(c_{n-k+j-i})_{1 \leq i, j \leq m-k},$$

where  $c_d$  is defined by

$$\begin{aligned} 1 + c_1 t + \dots + c_r t^r + \dots &= \frac{1}{(1-a_1 t)(1-a_2 t) \cdots (1-a_m t)} \\ &= (1 + a_1 t + a_1^2 t^2 + \dots) \cdots (1 + a_m t + a_m^2 t^2 + \dots), \end{aligned}$$

i.e.  $c_d = S_d(a_1, \dots, a_m)$ , the complete symmetric polynomial of degree  $d$  in the  $a_i$ .

In particular, if  $m \leq n$  and  $N - n + m - 1 \geq 0$ , then the degree of the determinantal variety defined by the maximal minors of  $M$  is

$$\deg Z_{m-1} = S_{n-m+1}(a_1, \dots, a_m) = \sum_{i_1+i_2+\dots+i_m=n-m+1} a_1^{i_1} \cdots a_m^{i_m} \quad (3.4)$$

### Symmetric determinantal varieties

Let  $M = (m_{ij})$  be a symmetric  $(n \times n)$ -matrix with homogeneous polynomial entries, and let  $Z_k$  be the projective variety defined by the  $(k+1) \times (k+1)$  minors of  $M$ , then the codimension of  $Z_k$  is at most  $\binom{n-k}{2}$ . The following proposition is obtained combining [3, Example 22.31] with Theorem 3.1.

**Proposition 3.2.** *Let  $M$  be a the symmetric  $(n \times n)$ -matrix  $M$  with general polynomial entries, then the subvariety  $Z_k$  where  $M$  has rank at most  $k$  has codimension equal to the minimum of  $\binom{n-k+1}{2}$  and  $N+1$ , and is singular precisely along  $Z_{k-1}$ .*

Harris and Tu's symmetric degeneracy formula, [2, Example 14.4.11], computes the degree of  $Z_k$ . When  $M$  is symmetric of dimension  $n$ , with general polynomial entries  $m_{ij} = m_{ji}$  of degree  $a_i + a_j$ , where  $a_1, \dots, a_n$  are positive half integers, then the degree of  $Z_k$ , if it is nonempty, is

$$\deg Z_k = 2^{n-k} \Delta_{(n-k, \dots, 2, 1)}(c) = 2^{n-k} \det \begin{pmatrix} c_{n-k} & c_{n-k+1} & \cdots & c_{2n-2k} \\ c_{n-k-2} & c_{n-k-1} & \cdots & c_{2n-2k-2} \\ c_{n-k-4} & c_{n-k-3} & \cdots & c_{2n-2k-4} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_1 \end{pmatrix},$$

where

$$1 + c_1 t + \cdots + c_r t^r + \cdots = (1 + a_1 t) \cdots (1 + a_n t).$$

In particular, if the entries are linear, i.e.  $a_i + a_j = 1$  for all index pairs  $(i, j)$ , then

$$\deg Z_k = \prod_{j=0}^{n-k-1} \frac{\binom{n+j}{n-k-j}}{\binom{2j+1}{j}}. \quad (3.5)$$

#### 3.3.4 Degree Formulas

##### 3.3.4.1 POP: The General Polynomial Optimization Problem

Consider the optimization problem

$$\begin{cases} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) = 0, i = 1, \dots, m \end{cases} \quad (3.6)$$

where the  $f_0(x), \dots, f_m(x)$  are general multivariate polynomial functions in  $\mathbb{R}[x]$ . It follows from Theorem 3.1 that  $\bar{K} = \bar{K}(f_1, \dots, f_m)$  is a smooth complete intersection. If in addition the level sets of the objective function are smooth at the critical points, then the KKT conditions are satisfied. For a general objective function  $f_0$  the number of critical points is then given by the cardinality of the intersection  $C(f_0) = C_{\bar{K}} \cap \bar{K}$ , where  $C_{\bar{K}}$  is the locus of points where the matrix (3.3) has rank at most  $m$ . Thus  $C_{\bar{K}}$  is a determinantal variety of degree given by the formula (3.4). Theorem 3.2 allows us to compute the cardinality of  $C(f_0)$  as the intersection  $C_{\bar{K}} \cap \bar{K}$ .

**Theorem 3.3 ([5, Theorem 2.2]).** *The algebraic degree of the general polynomial optimization problem (3.6) is equal to*

$$d_1 d_2 \cdots d_m S_{n-m}(d_0 - 1, d_1 - 1, \dots, d_m - 1),$$

where  $S_{n-m}$  is the complete symmetric polynomial of degree  $n-m$ .

### 3.3.4.2 Example

Consider the polynomial optimization problem

$$\begin{cases} \text{minimize} & x^2 + xy + y^2 + z^2 + z \\ \text{subject to} & x^3 + y^2z + x + z^3 + z = 0 \end{cases} \quad (3.7)$$

The homogenized objective and constraint functions are

$$F_0 = x^2 + xy + y^2 + z^2 + wz \quad \text{and} \quad F_1 = x^3 + y^2z + w^2x + z^3 + zw^2.$$

The constraint,  $F_1 = 0$ , defines a hypersurface  $\bar{K} \subset \mathbb{CP}^3$  of degree 3. For the critical locus we first compute the determinantal locus  $C_{\bar{K}}$ , defined by the  $(2 \times 2)$ -minors of the matrix

$$\begin{pmatrix} \frac{\partial}{\partial x} F_0 & \frac{\partial}{\partial x} F_1 \\ \frac{\partial}{\partial y} F_0 & \frac{\partial}{\partial y} F_1 \\ \frac{\partial}{\partial z} F_0 & \frac{\partial}{\partial z} F_1 \end{pmatrix} = \begin{pmatrix} 2x + y & 3x^2 + w^2 \\ x + 2y & 2yz \\ 2z + w & y^2 + 3z^2 + w^2 \end{pmatrix}$$

Since the columns in this  $(3 \times 2)$ -matrix are polynomials in degree 1 and 2, the formula (3.4) gives the degree  $S_2(1, 2) = 1^2 + 1 \cdot 2 + 2^2 = 7$  for the determinantal locus  $C_{\bar{K}}$ . The hypersurface  $\bar{K}$  has degree 3 and the critical locus  $\bar{K} \cap C_{\bar{K}}$  is a set of 21 points, whose  $z$ -coordinates are the zeros of an irreducible polynomial of degree 21 with first and last terms

$$1776384z^{21} + 8812800z^{20} + \cdots + 3848$$

in accordance with the above formula for the algebraic degree.

Adding one constraint function we get the problem

$$\begin{cases} \text{minimize} & x^2 + xy + y^2 + z^2 + z \\ \text{subject to} & x^3 + y^2z + x + z^3 + z = y^3 + xz + x^2z + 1 = 0 \end{cases} \quad (3.8)$$

The homogenized new constraint  $F_2 = y^3 + xzw + x^2z + w^3$ , defines together with  $F_1$  a complete intersection  $\bar{K} \subset \mathbb{CP}^3$  of codimension 2 and degree 9. For the critical locus we compute  $C_{\bar{K}}$ , defined by the determinant of the matrix

$$\begin{pmatrix} \frac{\partial}{\partial x} F_0 & \frac{\partial}{\partial x} F_1 & \frac{\partial}{\partial x} F_2 \\ \frac{\partial}{\partial y} F_0 & \frac{\partial}{\partial y} F_1 & \frac{\partial}{\partial y} F_2 \\ \frac{\partial}{\partial z} F_0 & \frac{\partial}{\partial z} F_1 & \frac{\partial}{\partial z} F_2 \end{pmatrix} = \begin{pmatrix} 2x+y & 3x^2+w^2 & 2xz+zw \\ x+2y & 2yz & 3y^2 \\ 2z+w & y^2+3z^2+w^2 & x^2+xw \end{pmatrix}$$

The degree of  $C_{\bar{K}}$  is 5, and the critical locus  $C_{\bar{K}} \cap \bar{K}$  is a set of  $9 \cdot 5 = 45$  points, whose  $z$ -coordinates are the zeros of an irreducible polynomial of degree 45 whose first and last terms are

$$3379071136256z^{45} + 10924921983488z^{44} + \cdots + 112940568.$$

Notice that if the constraints are

$$x^3 + y^2z + x = y^3 + xz + x^2z = 0,$$

then  $\bar{K}$  is still a complete intersection of degree 9, but the critical locus  $C_{\bar{K}} \cap \bar{K}$  contains the line  $x = y = 0$ , so it is not finite, and our formula does not apply. Restricted to this line, the objective function has one critical point. In the complement of this line  $C_{\bar{K}} \cap \bar{K}$  has 26 points.

### 3.3.4.3 QCQP: Quadratically Constrained Quadratic Programming

In the special case when all the polynomials  $f_0, f_1, \dots, f_m$  are quadratic, then problem (3.6) becomes a quadratically constrained quadratic programming problem which has the standard form

$$\begin{cases} \text{minimize} & x^T A_0 x + b_0^T x + c_0 \\ \text{subject to} & x^T A_i x + b_i^T x + c_i \geq 0, i = 1, \dots, \ell. \end{cases} \quad (3.9)$$

Here  $A_i, b_i, c_i$  are matrices or vectors of appropriate dimensions. The objective function and all the constraints are quadratic polynomials. Suppose  $m \leq \ell$  of the constraints are equalities at an optimal solution. Then, by Theorem 3.3, the algebraic degree of the optimization problem (3.9) is

$$2^m \cdot S_{n-m}(1, \underbrace{1, \dots, 1}_{m \text{ times}}) = 2^m \cdot \sum_{i_0+i_1+i_2+\dots+i_m=n-m} 1 = 2^m \cdot \binom{n}{m}. \quad (3.10)$$

### 3.3.4.4 SDP: Semidefinite Programming

In semi-definite programming one optimizes an objective function over the positive semidefinite matrices in an affine space of symmetric matrices.

Consider the optimization problem

$$\begin{cases} \text{maximize} & \text{trace}(B \cdot Y) \\ \text{subject to} & Y \in \mathcal{U} \\ \text{and} & Y \geq 0. \end{cases} \quad (3.11)$$

where  $B$  is a real symmetric  $n \times n$ -matrix,  $\mathcal{U}$  is a  $m$ -dimensional affine subspace in the  $\binom{n+1}{2}$ -dimensional space of real  $n \times n$ -symmetric matrices, and  $Y \geq 0$  means that  $Y$  is positive semidefinite (all  $n$  eigenvalues of  $Y$  are non-negative). The problem is called *feasible* if  $\mathcal{U}$  contains at least one positive definite matrix, and we assume this natural condition from here on.

The rank defines a stratification of  $\mathcal{U} = \mathcal{U}_1 \cup \mathcal{U}_2 \cup \dots \cup \mathcal{U}_n$  such that the closure of the stratum  $\mathcal{U}_r$  equals  $\cup_{k \leq r} \mathcal{U}_r$  and is a symmetric determinantal variety. The formula (3.5) of Harris and Tu, gives the degree of the closure of each stratum  $\mathcal{U}_i$ .

By Proposition 3.2, this stratification is an example of the decomposition of  $K$  described in Sect. 3.2. We may consider the critical set of problem (3.11) on each of the strata. It is therefore reasonable to find an algebraic degree for each rank  $r$ .

Pataki's inequalities says that for a general space of matrices  $\mathcal{U}$ , the rank of an optimal solution lies in a certain range:

**Proposition 3.3.** (Pataki's Inequalities [8, Corollary 3.3.4])

Let  $r$  be the rank of the optimal matrix  $\hat{Y}$  of the optimization problem (3.11). Then

$$\binom{n-r+1}{2} \leq m \quad \text{and} \quad \binom{r+1}{2} \leq \binom{n+1}{2} - m. \quad (3.12)$$

For each rank  $r$  in this range there is an open set of initial data  $B$  and  $\mathcal{U}$  for which the optimal matrix have this rank.

These inequalities reflect, on the one hand, that when  $m$  is small compared to  $n - r$ , the stratum of rank  $r$ -matrices  $\mathcal{U}_r$  is empty, and, on the other hand, when  $r$  and  $m$  are big compared to  $n$ , the general linear objective function has no critical points on  $\mathcal{U}_r$  (only at matrices of smaller rank).

In the range given by the Pataki inequalities it makes sense to ask for a general formula for the algebraic degree of an optimal solution. In fact, the general formula for the algebraic degree  $\delta(m, n, r)$  of the optimization problem (3.11) depends precisely on the three parameters  $m, n, r$  [4, Theorem 1.1]. It is expressed in terms of a function  $\psi_I$  on subsequences  $I$  of  $\{1, \dots, n\}$ . To define this function, consider the matrix of binomial coefficients

$$\Psi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \dots \\ 1 & 1 & 0 & 0 & 0 & 0 & \dots \\ 1 & 2 & 1 & 0 & 0 & 0 & \dots \\ 1 & 3 & 3 & 1 & 0 & 0 & \dots \\ 1 & 4 & 6 & 4 & 1 & 0 & \dots \\ 1 & 5 & 10 & 10 & 5 & 1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

and define  $\psi_I$ , where  $I = \{i_1, \dots, i_r\} \subset \{1, \dots, n\}$ , as the sum of the  $r \times r$  minors of the submatrix  $\Psi_I \subset \Psi$  consisting of the rows  $i_1, \dots, i_r$ . A different, but equivalent, definition of  $\psi_I$  goes as follows. Let

$$\psi_i = 2^{i-1}, \quad \psi_{i,j} = \sum_{k=i}^{j-1} \binom{i+j-2}{k} \quad \text{when } i < j,$$

then

$$\psi_{i_1, \dots, i_r} = \text{Pf}(\psi_{i_k, l})_{1 \leq k < l \leq r} \quad \text{if } r \text{ is even}$$

$$\psi_{i_1, \dots, i_r} = \text{Pf}(\psi_{i_k, l})_{0 \leq k < l \leq r} \quad \text{if } r \text{ is odd}$$

where  $\psi_{i_0, i_k} = \psi_{i_k}$  and Pf denotes the Pfaffian.

**Theorem 3.4 ([4, Theorem 1.1]).** *The algebraic degree of the semidefinite programming problem (3.11) equals*

$$\delta(m, n, r) = \sum_I \psi_I \psi_{I^c} \tag{3.13}$$

where the sum runs over all strictly increasing subsequences  $I = \{i_1, \dots, i_{n-r}\}$  of  $\{1, \dots, n\}$  of length  $n-r$  and sum  $i_1 + \dots + i_{n-r} = m$ , and  $I^c$  is the complement  $\{1, \dots, n\} \setminus I$ .

Notice that the formula makes sense only when the Pataki inequalities  $m \geq \binom{n-r+1}{2}$  and  $\binom{r+1}{2} \leq \binom{n+1}{2} - m$  are satisfied.

### 3.3.4.5 Example

Let  $B, A_1, \dots, A_{10}$  be general symmetric  $6 \times 6$  matrices, then problem (3.11) is solved finding critical points on each stratum of the boundary of the positive semidefinite matrices in the affine space  $\mathcal{U} = \{A(x) = x_1 A_1 + \dots + x_{10} A_{10}; x \in \mathbb{R}^{10}\}$ . The boundary, of the set of positive semidefinite matrices, consists of matrices with some vanishing eigenvalues. The natural stratification of the boundary, by the dimension of the eigenspace of the eigenvalue 0, is precisely the rank stratification of the  $\mathcal{U}$ , i.e. each stratum consists of the matrices of a fixed rank. Notice that, by Proposition 3.2 and Theorem 3.1, these strata coincide with the strata of singular sets as

defined above. Of course, only part of each rank stratum lies on the boundary of the positive semidefinite matrices where no eigenvalue is negative, but the Zariski closure will in general contain the whole rank stratum, so for our computations the rank stratification is the relevant stratification. The set of matrices in  $\mathcal{U}$  of rank  $6-r$  has codimension  $\binom{r+1}{2}$ , so the rank stratification has the following nonempty strata:

$$\mathcal{U} = \mathcal{U}_6 \cup \mathcal{U}_5 \cup \mathcal{U}_4 \cup \mathcal{U}_3 \cup \mathcal{U}_2$$

By formula (3.5) the degrees of the closure of the strata of singular matrices are

$$\deg \bar{\mathcal{U}}_5 = 6, \quad \deg \bar{\mathcal{U}}_4 = 35, \quad \deg \bar{\mathcal{U}}_3 = 112 \quad \text{and} \quad \deg \bar{\mathcal{U}}_2 = 126.$$

The Zariski closure of the boundary of the set of positive semidefinite matrices is  $\mathcal{U}_5 \cup \mathcal{U}_4 \cup \mathcal{U}_3 \cup \mathcal{U}_2$ . The critical locus of the linear function  $\text{trace}(B \cdot A(x))$  is by the genericity assumption finite in each stratum and of cardinality given by the formula (3.13). Thus the cardinality of the critical locus of  $\text{trace}(B \cdot A(x))$  on  $\mathcal{U}_k$  is  $\delta(10, 6, k)$ .

First, notice that  $\delta(10, 6, 5) = 0$ , since  $\binom{5+1}{2} > \binom{6+1}{2} - 10$ , i.e. the Pataki inequality is violated. This simply means that although  $\bar{\mathcal{U}}_5$  is a hypersurface of degree 6 the general objective function does not have any critical points outside its singular locus. In fact, any objective function that has a critical locus at a nonsingular point of  $\bar{\mathcal{U}}_5$ , will have critical locus containing a linear subspace in this hypersurface.

For the smaller ranks the formula says

$$\begin{aligned}\delta(10, 6, 4) &= \psi_{4,6}\psi_{1,2,3,5}, \\ \delta(10, 6, 3) &= \psi_{1,3,6}\psi_{2,4,5} + \psi_{1,4,5}\psi_{2,3,6} + \psi_{2,3,5}\psi_{1,4,6} \\ \delta(10, 6, 2) &= \psi_{1,2,3,4}\psi_{5,6}.\end{aligned}$$

We consider

$$\Psi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 \\ 1 & 3 & 3 & 1 & 0 & 0 \\ 1 & 4 & 6 & 4 & 1 & 0 \\ 1 & 5 & 10 & 10 & 5 & 1 \end{bmatrix},$$

and compute the  $\psi_{i_1, \dots, i_r}$ , with  $r = 1, 2, 3$  and  $I = \{i_1, \dots, i_r\} \subset \{1, \dots, 6\}$ , as the sum of the  $r \times r$  minors of the submatrix  $\mathbb{P}si_I \subset \mathbb{P}si$  consisting of the rows  $i_1, \dots, i_r$ . Thus

$$\psi_1 = 1, \psi_2 = 2, \psi_3 = 4, \psi_4 = 8, \psi_5 = 16, \psi_6 = 32$$

$$\psi_{1,2} = 1, \psi_{1,3} = 3, \psi_{1,4} = 7, \psi_{1,5} = 15, \psi_{1,6} = 31$$

$$\psi_{2,3} = 3, \psi_{2,4} = 10, \psi_{2,5} = 25, \psi_{2,6} = 56,$$

$$\psi_{3,4} = 31, \psi_{3,5} = 35$$

To illustrate the definition using Pfaffians, we compute a value using that definition:

$$\begin{aligned}\psi_{3,6} &= \binom{7}{3} + \binom{7}{4} + \binom{7}{5} = 91, \\ \psi_{4,5} &= \binom{7}{4} = 35, \psi_{4,6} = \binom{8}{4} + \binom{8}{5} = 126, \\ \psi_{5,6} &= \binom{9}{5} = 126, \\ \psi_{1,3,6} &= \text{Pf} \begin{bmatrix} 0 & \psi_1 & \psi_3 & \psi_6 \\ -\psi_1 & 0 & \psi_{1,3} & \psi_{1,6} \\ \psi_2 & -\psi_{1,3} & 0 & \psi_{3,6} \\ \psi_6 & -\psi_{1,6} & -\psi_{3,6} & 0 \end{bmatrix} \\ &= \psi_1 \psi_{3,6} - \psi_3 \psi_{1,6} + \psi_{1,3} \psi_6 = 1 \cdot 91 - 4 \cdot 31 + 3 \cdot 32 = 65.\end{aligned}$$

Furthermore

$$\begin{aligned}\psi_{1,4,6} &= \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 3 \\ 1 & 5 & 10 \end{bmatrix} + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 1 \\ 1 & 5 & 10 \end{bmatrix} + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 5 & 5 \end{bmatrix} \\ &\quad + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 5 & 1 \end{bmatrix} + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 1 \\ 1 & 10 & 10 \end{bmatrix} + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 10 & 5 \end{bmatrix} \\ &\quad + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 10 & 1 \end{bmatrix} + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 10 & 5 \end{bmatrix} + \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 10 & 1 \end{bmatrix} \\ &= 15 + 25 + 15 + 3 + 20 + 15 + 3 + 5 + 1 = 102\end{aligned}$$

$$\psi_{1,4,5} = 27, \psi_{2,3,5} = 18, \psi_{2,4,5} = 30, \psi_{2,3,6} = 54$$

$$\psi_{1,2,3,4} = \det \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} = 1, \quad \psi_{1,2,3,5} = \det \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 4 & 6 & 4 \end{bmatrix} + \det \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 4 & 6 & 1 \end{bmatrix} = 5$$

Thus

$$\delta(10, 6, 4) = \psi_{4,6} \cdot \psi_{1,2,3,5} = 141 \cdot 5 = 705,$$

$$\delta(10, 6, 3) = \psi_{1,3,6} \cdot \psi_{2,4,5} + \psi_{1,4,5} \cdot \psi_{2,3,6} + \psi_{2,3,5} \cdot \psi_{1,4,6}$$

$$\begin{aligned}
 &= 65 \cdot 30 + 27 \cdot 54 + 18 \cdot 102 = 5244, \\
 \delta(10, 6, 2) &= \psi_{1,2,3,4} \cdot \psi_{5,6} = 1 \cdot 126 = 126.
 \end{aligned}$$

Notice that  $\mathcal{U}_2$  is finite, so  $\delta(10, 6, 2)$  equals the degree of  $\mathcal{U}_2$  computed above.

## References

1. Cox, D., Little, J., O’Shea, D.: Ideals, Varieties and Algorithms. UTM, 3rd edition, Springer-Verlag, New York (2007)
2. Fulton, W.: Intersection Theory. Springer-Verlag, New York (1984)
3. Harris, J.: Algebraic Geometry, A First Course. Springer-Verlag, New York (1992)
4. Graf von Bothmer, H.C., Ranestad, K.: A general formula for the algebraic degree in semidefinite programming. Bull. Lond. Math. Soc. **41**, 193–197 (2009)
5. Nie, J., Ranestad, K.: The algebraic degree of polynomial optimization. SIAM J. Optim. **20**, 485–502 (2009)
6. Nie, J., Ranestad, K., Sturmfels, B.: The algebraic degree of semidefinite programming. Mathematical Programming A **122**, 379–405 (2010)
7. Nocedal, J., Wright, S.: Numerical Optimization. Springer Series in Operations Research, Springer-Verlag, New York (1999)
8. Pataki, G.: The geometry of cone-LP’s. In: Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.) Handbook of Semidefinite Programming, pp. 29–66. Kluwer (2000)

# Chapter 4

## Semidefinite Representation of Convex Sets and Convex Hulls

J. William Helton and Jiawang Nie

### 4.1 Introduction

A set  $S$  is said to have an LMI representation, or to be LMI representable if

$$S = \{x \in \mathbb{R}^n : A_0 + A_1 x_1 + \cdots + A_n x_n \succeq 0\} \quad (4.1)$$

for some symmetric matrices  $A_i$ . If so, we just say  $S$  has an LMIr or is LMIr. Here the notation  $X \succeq 0 (> 0)$  means the matrix  $X$  is positive semidefinite (definite).

If  $S$  in (4.1) has an interior point, then  $A_0$  can be assumed to be positive definite (e.g.,  $A_0 = I$ ), without loss of generality; when  $A_0 = I$ , (4.1) is called a monic LMI representation. Clearly, to have an LMIr,  $S$  must be convex.

Our main interest here is how to represent a convex set  $S$  or the convex hull of a nonconvex set  $S$  with an LMI; or if this is not possible, replace  $S$  with a “useful” set  $\hat{S}$  which has an LMIr. In particular, we treat what are called semidefinite programming (SDP) representations. The chapter addresses theory and constructions and gives examples.

#### 4.1.1 LMI Representations

Section 4.2 sketches what is known about which sets  $S \subset \mathbb{R}^n$  have a monic LMI representation. As we shall see such  $S$  must be convex, basic semialgebraic, and pass a so-called “line test” (rigid convexity).

---

J.W. Helton (✉) • J. Nie

Department of Mathematics, University of California at San Diego, 9500 Gilman Drive,  
La Jolla, CA 92093-0112, USA

e-mail: [helton@math.ucsd.edu](mailto:helton@math.ucsd.edu); [njw@math.ucsd.edu](mailto:njw@math.ucsd.edu)

Often for linear engineering systems the unknowns  $X_j$  are themselves square matrices. One has polynomials in non-commuting variables, and LMIs with matrix variables  $X_j$  entering the tensor products. This is the beginning of a new subject: non-commutative real algebraic geometry. The Chap. 13 of this Handbook surveys this topic.

### 4.1.2 SDP Representations

It is known that not every convex semialgebraic set is representable by LMI. For instance, the convex set

$$T = \left\{ x \in \mathbb{R}^2 : 1 - (x_1^4 + x_2^4) \geq 0 \right\} \quad (4.2)$$

does not admit an LMIR [11], since it is not rigidly convex. However, the set  $T$  is the projection into  $x$ -space of the set

$$\hat{S} := \left\{ (x, w) \in \mathbb{R}^2 \times \mathbb{R}^2 : \begin{bmatrix} 1+w_1 & w_2 \\ w_2 & 1-w_1 \end{bmatrix} \succeq 0, \begin{bmatrix} 1 & x_1 \\ x_1 & w_1 \end{bmatrix} \succeq 0, \begin{bmatrix} 1 & x_2 \\ x_2 & w_2 \end{bmatrix} \succeq 0 \right\}$$

in  $\mathbb{R}^4$  which is represented by an LMI.

If the convex set  $S$  can be represented as the projection to  $\mathbb{R}^n$  of a set  $\hat{S}$  in a higher dimensional space  $\mathbb{R}^{n+N}$  which has an LMI representation, then  $S$  is called semidefinite programming representable or SDP representable. If so, we also just say  $S$  is SDr or has an SDr. Sometimes we refer to a semidefinite representation as a lifted LMI representation of the convex set  $S$  and to the LMI in (4.3) as a lifted LMI for  $S$ , and to  $\hat{S}$  as the SDP lift of  $S$ . Expressed algebraically, the lift  $\hat{S}$  has the form

$$\hat{S} = \left\{ (x, u) \in \mathbb{R}^{(n+N)} : A_0 + \sum_{i=1}^n A_i x_i + \sum_{j=1}^N B_j u_j \succeq 0 \right\} \subset \mathbb{R}^{(n+N)}, \quad (4.3)$$

that is  $S = \{x \in \mathbb{R}^n : \exists u \in \mathbb{R}^N, (x, u) \in \hat{S}\}$ , for some symmetric matrices  $A_i$  and  $B_j$ .

Section 4.3 sketches what is known about which sets have an SDr. It is obvious that, to have an SDr  $S$  must be convex and semialgebraic. We give sufficient conditions for SDr which are not enormously more stringent. A summary of the main results for SDP representability is:

**Theorem 4.1 (The Main SDP Representability Theorem).** *Let  $S$  be a compact convex semialgebraic set having nonempty interior. If each irreducible piece of the boundary of  $S$  is “nonsingular” and has “positive curvature”, then  $S$  is SDP representable. This sufficiency condition is not far away from being necessary, because every essential piece of the boundary of a convex set has nonnegative curvature on its smooth parts.*

Precise definitions of conditions in the above are found in Sect. 4.3, and a rigorous version of this statement is contained in Theorems 4.6, 4.8, 4.9.

### 4.1.3 Convex Hulls of Semialgebraic Sets

The problem of characterizing convex hulls of algebraic or semialgebraic sets is being seriously pursued currently. Henrion [12] discussed the convex hull of rational varieties, Ranestad and Sturmfels [24, 25] discussed convex hulls of space curves and varieties, Sanyal et al. [27] discussed convex hulls of orbits of compact groups acting linearly on vector spaces, and Scheiderer [28] discussed the convex hulls of curves of genus one.

In this chapter, we focus on convex hulls of bounded nonconvex semialgebraic sets having nonempty interiors. For such nonconvex sets, there is an appealing theory, including a formula for representing the convex hull of a union of SDP representable sets, a localization procedure which follows from this, and a pleasing sufficient condition (near to being necessary) on semialgebraic sets to have a SDP representable convex hull. This is mainly described in Sect. 4.7.

### 4.1.4 Constructions and Theory

Basic to SDP representation is the Lasserre type relaxation and a construction by Lasserre and independently Parrilo. This is the only systematic way currently known to produce SDP lifts. It associates to a semialgebraic set  $S$  in  $\mathbb{R}^n$  a hierarchy of sets  $S_{n+k}$  in increasingly high dimensional space. The projections into  $\mathbb{R}^n$  of each of them contain  $S$ , and as  $k$  increases these projections are closer to  $S$ . The basic question is: does this process stop for some finite  $k$ ?

We shall present constructions for special situations that might be practically implementable. They are critical to the general theory. The constructions are in Sect. 4.4 and the theory is in Sects. 4.6 and 4.7. Theorem 4.14 is new in that it validates a more concrete construction than our earlier work.

### 4.1.5 Motivation

The key use of SDP representations is illustrated by optimizing a linear function  $\ell^T x$  over  $S$ . Note that minimizing  $\ell^T x$  over  $S$  is equivalent to problem

$$\min_{(x,y) \in \hat{S}} \ell^T x,$$

which is a standard SDP, so can be solved by standard SDP software. Nesterov and Nemirovski [17] in their book gave collections of examples of SDP representable sets, thereby leading to the representation problems we are addressing here.

#### 4.1.6 Notations

The symbol  $\mathbb{N}$  (resp.,  $\mathbb{R}$ ,  $\mathbb{C}$ ) denotes the set of nonnegative integers (resp., real numbers, complex numbers), and  $\mathbb{R}_+^n$  denotes the nonnegative orthant of  $\mathbb{R}^n$ . The  $e_i$  denotes the  $i$ -th standard unit 1/0 vector. The  $\Delta_m = \{\lambda \in \mathbb{R}_+^m : \lambda_1 + \dots + \lambda_m = 1\}$  denotes the standard simplex. For  $x \in \mathbb{R}^n$ ,  $\|x\| = \sqrt{\sum_{i=1}^n x_i^2}$ .  $B(u, r)$  denotes the open ball  $\{x \in \mathbb{R}^n : \|x - u\| < r\}$  and  $\bar{B}(u, r)$  denotes the closed ball  $\{x \in \mathbb{R}^n : \|x - u\| \leq r\}$ . For  $\alpha \in \mathbb{N}^n$ ,  $|\alpha| := \alpha_1 + \dots + \alpha_n$ ,  $x^\alpha := x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ . For any  $t \in \mathbb{R}$ ,  $\lceil t \rceil$  denotes the smallest integer not smaller than  $t$ .

For  $x \in \mathbb{R}^n$ ,  $x_i$  denotes the  $i$ -th component of  $x$ , that is,  $x = (x_1, \dots, x_n)$ . For  $\alpha \in \mathbb{N}^n$ , denote  $|\alpha| = \alpha_1 + \dots + \alpha_n$ . For  $x \in \mathbb{R}^n$  and  $\alpha \in \mathbb{N}^n$ ,  $x^\alpha$  denotes  $x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ . The symbol  $[x]_d$  denotes the following vector of monomials

$$[x]_d^T = \begin{bmatrix} 1 & x_1 & \cdots & x_n & x_1^2 & x_1 x_2 & \cdots & x_1^d & x_1^{d-1} x_2 & \cdots & x_n^d \end{bmatrix},$$

The  $[x^d]$  denotes the column vector of all monomials of degree  $d$ , i.e.,

$$[x^d]^T = \begin{bmatrix} x_1^d & x_1^{d-1} x_2 & \cdots & x_n^d \end{bmatrix}.$$

The symbol  $\mathbb{R}[x] = \mathbb{R}[x_1, \dots, x_n]$  (resp.  $\mathbb{C}[x] = \mathbb{C}[x_1, \dots, x_n]$ ) denotes the ring of polynomials in  $(x_1, \dots, x_n)$  with real (resp. complex) coefficients. A polynomial is called a form if it is homogeneous.  $\mathbb{R}[x]_{\leq d}$  denotes the subspace of polynomials in  $\mathbb{R}[x]$  of degree at most  $d$ .

A polynomial  $p$  is said to be a sum of squares (SOS) if  $p(x) = w(x)^T w(x)$  for some column vector polynomial  $w \in \mathbb{R}[x]^p$  with some  $p$ . A matrix polynomial  $H$  is said to be SOS if  $H(x) = W(x)^T W(x)$  for some matrix polynomial  $W \in \mathbb{R}[x]^{p \times r}$  with some  $p$  and  $r$ . For a given set  $W$ ,  $\overline{W}$  denotes the closure of  $W$ ,  $\partial W$  denotes its topological boundary, and  $\text{int}(W)$  denotes its interior.

For a given matrix  $A$ ,  $A^T$  denotes its transpose.  $I_n$  denotes the  $n \times n$  identity matrix. For a symmetric matrix  $X$ ,  $X \geq 0$  (resp.,  $X > 0$ ) means  $X$  is positive semidefinite (resp. positive definite). For a symmetric matrix  $A$ ,  $\lambda_{\min}(A)$  denotes the smallest eigenvalue of  $A$ , and  $\|A\|_2$  denotes the standard 2-norm of  $A$ , and  $\|A\|_F$  denotes the Frobinus norm of  $A$ , i.e.,  $\|A\|_F = \sqrt{\text{Trace}(A^T A)}$ . Recall that  $\|A\|_2 \leq \|A\|_F$ .

## 4.2 LMI Representation

A matrix valued function of the form

$$L(x) := A_0 + A_1 x_1 + \cdots + A_n x_n \quad (4.4)$$

is called a linear pencil and is said to be monic if  $A_0 = I$ . A set  $S \subset \mathbb{R}^n$  has a linear matrix inequality representation or LMI representation provided there is a linear pencil  $L(x)$  for which

$$S = \{x \in \mathbb{R}^n : L(x) \succeq 0\}.$$

If so, we say  $S$  has an LMIr or is LMIr. We assume that the set  $S$  has a nonempty interior, and take the pencil to be monic, e.g.,  $L(0) = I$ . Parrilo and Sturmfels (see [22]) formally ask:

*Question 4.1. Which sets have a monic LMI representation?*

LMI representability is closely related to determinantal representation. A polynomial  $\check{p}$  has a monic determinantal representation if

$$\check{p}(x) = \kappa \det[I + A_1 x_1 + \cdots + A_n x_n]. \quad (4.5)$$

where the  $A_j$  are symmetric  $d \times d$  matrices, and  $\kappa$  is a constant. This representation is said to be a monic determinantal representation for  $\check{p}$  and  $d$  is called the size of the representation. The following question is related to Question 4.1.

*Question 4.2. Which polynomials have a monic determinantal representation?*

A version of this question for homogeneous polynomials was posed already by Lax [15] in 1958.

As we shall see an answer to Q 4.2 also answers Q 4.1. This section describes work in [11] which settles the questions in the two dimensional case ( $n = 2$ ).

### 4.2.1 Necessary Conditions

Suppose a monic pencil  $L(x)$  represents a set  $S$  as in (4.1). Clearly, it holds that:

- (i)  $S$  is a convex set (with 0 being an interior point of  $S$ ).
- (ii) The polynomial  $\check{p}(x) := \det(L(x))$  is positive on the interior of  $S$  and vanishes on its boundary  $\partial S$ . The  $\partial S$  lies on the zero set of the polynomial  $\check{p}(x) := \det L(x)$ , so  $\partial S$  lies on an algebraic variety.
- (iii)  $S$  must be a basic closed semialgebraic set, i.e.,

$$S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}$$

where  $g_i(x)$  are polynomials. For instance, the  $g_i$  would be chosen as the principle minors of  $L(x)$ .

This leads us to formalize a key property of sets which appear throughout this subject. A closed subset  $S$  of  $\mathbb{R}^n$  is called an algebraic interior if it equals the closure of a (arcwise) connected component of  $\{x \in \mathbb{R}^n : p(x) > 0\}$  for some polynomial  $p$  in  $x$ . In this case  $p$  is called a defining polynomial for  $S$ . If  $d$  denotes the degree of  $p$ , we say that  $S$  is an algebraic interior of degree  $d$ . Clearly, if a set  $S$  is represented by a monic pencil  $L(x)$ , then  $S$  is an algebraic interior with  $\check{p}$  as a defining polynomial.

The following lays out very nice properties of algebraic interiors.

**Theorem 4.2.** *A minimum degree defining polynomial  $p$  for an algebraic interior  $S$  is unique (up to a positive constant factor), and divides any other defining polynomial  $q$  for  $S$ , that is,  $q = fp$  with  $f$  a polynomial. The zero variety of  $p$  is the Zariski closure of the boundary of the algebraic interior  $S$  and is the union*

$$Z_1 \cup Z_2 \cup \cdots \cup Z_m \quad (4.6)$$

of irreducible varieties, each of real codimension 1.

*Proof.* The uniqueness of  $p$  is shown in Lemma 2.1 [11], while the decomposition is proved in its proof.  $\square$

#### 4.2.1.1 Real Zero Condition

For a monic determinantal representation to exist, there is an obvious necessary condition. Observe from (4.5) that

$$\check{p}(\mu x) := \mu^d \det \left[ \frac{I}{\mu} + A_1 x_1 + \cdots + A_n x_n \right].$$

For any fixed real vector  $x$ , since all eigenvalues of the symmetric matrix  $A_1 x_1 + \cdots + A_n x_n$  are real, the polynomial  $p(\mu x)$  in  $\mu$  vanishes only at real points. This condition is critical enough that we formalize it in a definition.

A real polynomial  $p$  satisfies the Real Zero (RZ) condition if for every  $x \in \mathbb{R}^n$  the polynomial  $p_x(\mu) = p(\mu x)$  in  $\mu$  has only real zeros. A Real Zero (RZ) polynomial is a polynomial satisfying the RZ condition. There is of course a similar RZ condition  $RZ_{x^0}$  with respect to any given point  $x^0 \in \mathbb{R}^n$ , where we consider the zeroes of  $p(x^0 + \mu x)$  for each  $x \in \mathbb{R}^n$ . As noticed by Lewis et al. [16], RZ polynomials are simply the nonhomogeneous version of the hyperbolic polynomials that were first introduced by Petrowski and Garding in the study of PDEs.

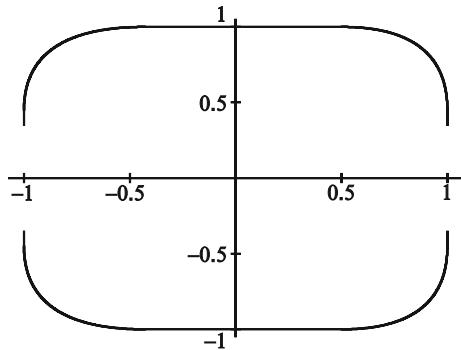
*Example 4.1.* For  $p(x) := 1 - (x_1^4 + x_2^4)$  and  $x = (x_1, x_2) \in \mathbb{R}^2$ ,

$$p(\mu x) = \left[ 1 - \mu^2 (x_1^4 + x_2^4)^{\frac{1}{2}} \right] \left[ 1 + \mu^2 (x_1^4 + x_2^4)^{\frac{1}{2}} \right]$$

has two complex (not real) zeroes  $\mu_{\pm} := i(x_1^4 + x_2^4)^{-\frac{1}{2}}$ . Thus  $p$  does not satisfy the RZ condition.

**Fig. 4.1**

$$p(x_1, x_2) = 1 - x_1^4 - x_2^4$$



#### 4.2.1.2 Rigid Convexity

Consider an algebraic interior  $S$  of degree  $d$  in  $\mathbb{R}^n$  with minimum degree defining polynomial  $p$ . Then  $S$  is called rigidly convex if for some fixed point  $x^0$  in the interior of  $S$  almost every line  $\ell$  through  $x^0$  intersects the real hypersurface  $p(x) = 0$  in exactly  $d$  points.<sup>1</sup> If this is the case, we say  $S$  passes the line test.

**Proposition 4.1** (see [11]).

1. An algebraic interior which passes the line test is a convex set.
2. If the line test defining rigid convexity holds for one point  $x^0$  in the interior of  $S$ , then it holds for all points in the interior of  $S$ .
3. Rigid convexity of an algebraic interior  $S$  containing 0 as an interior point is the same as its minimal degree defining polynomial  $p$  having the RZ Property.

*Example 4.2 (Example 4.1 Revisited).* Consider  $p(x_1, x_2) = 1 - x_1^4 - x_2^4$ . The convex algebraic interior  $S := \{x : p(x) \geq 0\}$  has degree 4 (since  $p$  is irreducible, it is a minimum degree defining polynomial), but all lines in  $\mathbb{R}^2$  through an interior point intersect the set  $p = 0$  in exactly two places. Thus  $S$  is not rigidly convex (Fig. 4.1).

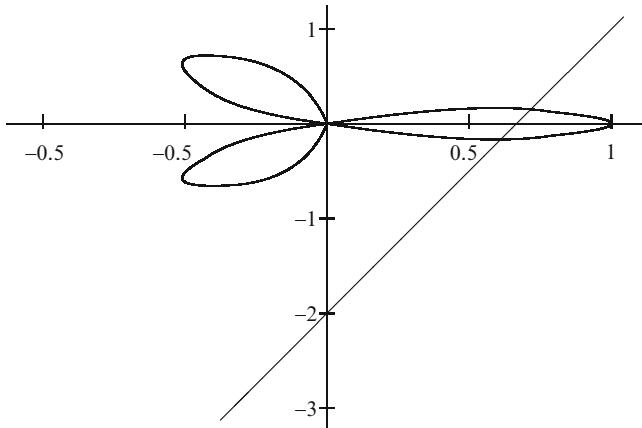
*Example 4.3.* Figure 4.2 show the zero set of  $p(x_1, x_2) = x_1^3 - 3x_2^2x_1 - (x_1^2 + x_2^2)^2$ . The complement to  $p = 0$  in  $\mathbb{R}^2$  consists of four components, three bounded convex components where  $p > 0$  and an unbounded component where  $p < 0$ . Let us analyze one of the bounded components, say the one in the right half plane,  $S$  is the closure of

$$\{x : p(x) > 0, x_1 > 0\}.$$

Is  $S$  rigidly convex? To check this: fix a point  $O$  inside  $S$ , e.g.  $O = (.7, 0)$ .

---

<sup>1</sup>One can replace here “almost every line” by “every line” if one counts multiplicities into the number of intersections, and also counts the intersections at infinity, i.e., replaces the affine real algebraic hypersurface  $p(x) = 0$  in  $\mathbb{R}^n$  by its projective closure in  $\mathbb{P}^n(\mathbb{R})$ .



**Fig. 4.2** A line thru  $O = (0, 0.7)$  hitting  $x_1^3 - 3x_2^2x_1 - (x_1^2 + x_2^2)^2 = 0$  in only 2 points

We can see from the picture that there is a continuum of real lines  $\ell$  through  $(0, 0.7)$  intersecting  $p = 0$  in exactly two real points. Thus  $S$  is not rigidly convex.

#### 4.2.2 The Main Result About LMI Representability

**Theorem 4.3.** *If  $S$  is LMI representable, then  $S$  passes the line test, i.e., it is rigidly convex. The converse is true in the two dimensional case. Furthermore, in this case there exists a LMI representation of the size equal to the degree of the boundary  $\partial S$ .*

The direct implication is clear in view of the previous subsection: if  $S$  is represented by a monic pencil  $L(x)$  then  $S$  is an algebraic interior with defining polynomial  $\check{p}(x) = \det L(x)$  which satisfies the RZ condition; therefore the minimum degree defining polynomial  $p$  of  $S$  is a factor of  $\check{p}$  (by Theorem 4.2). Thus  $p$  satisfies the RZ condition, so  $S$  is rigidly convex.

The converse is established by showing that

**Theorem 4.4.** *When  $n=2$ , any RZ bivariate polynomial of degree  $d$  admits a monic determinantal representation of size  $d$ .*

The proof uses classical one-dimensional algebraic geometry on the complexification of the projective closure of the real algebraic curve  $p(x) = 0$ . There are two possible constructions, one using theta functions and the Abel–Jacobi map, and one using bases of certain linear systems, more concretely, curves of degree  $d - 1$  that are everywhere tangent to the original curve. The actual computational aspects of both constructions remain to be explored.

*Conjecture 4.1.* For any dimension  $n$ , if  $S$  is rigidly convex, then  $S$  has a LMI representation.

When  $n > 2$ , the minimal size of a LMI representation in Conjecture 4.1 is in general larger than the degree of  $S$  if it is true.

### 4.2.3 Determinantal Representations

Now we turn to Q 4.2. The good news is

**Theorem 4.5 ([10]).** Every polynomial  $\check{p}$  (with  $\check{p}(0) \neq 0$ ) admits a symmetric determinantal representation,

$$\check{p}(x) = \kappa \det[J + A_1 x_1 + \cdots + A_n x_n], \quad (4.7)$$

with  $J$  a “signature matrix”, that is,  $J$  is symmetric and  $J^2 = I$ ,  $A_j$  are symmetric matrices, and  $\kappa$  is a constant.

The proof of Theorem 4.5 in [10] uses *noncommutative techniques* like those in this volume. First, one proves that every symmetric noncommutative polynomial  $p$  (with  $p(0) \neq 0$ ) admits a noncommutative symmetric determinantal representation just like the one above. This implies immediately Theorem 4.5 by taking  $p$  to be an arbitrary symmetric noncommutative polynomial whose “commutative collapse” is the given commutative polynomial  $\check{p}$ , and substituting scalar (rather than matrix) variables.

Of course, the symmetric determinantal representation in Theorem 4.5 is not monic, i.e.,  $J \neq I$ . One might hope that: every RZ polynomial has a monic determinantal representation? This was conjectured to be true in [11] and called the generalized Lax conjecture drawing on Lewis et al. [16] who settled a conjecture raised by Peter Lax in 1958 affirmatively for  $n=2$ , using Theorem 4.4. They observe that the Lax conjecture is false for  $n>2$ . Branden in [4] produced a counterexample to the generalized Lax conjecture: not every RZ polynomial has a monic determinantal representation. This may seem inconsistent with Conjecture 4.1, but it only amounts to requiring one (not all) defining polynomial  $\check{p}$  for  $S$  have a determinantal representation.

Here we point out a necessary condition on  $\check{p}$  for it to have a monic determinantal representation  $\check{p}(x) = \det(L(x))$ . Upon showing this to Peter Branden he proved that the property of  $\check{p}$  concluded in Proposition 9 only requires that  $\check{p}$  be a real zero polynomial.

**Proposition 4.2.** Suppose  $\check{p}$  has a monic determinantal representation  $\check{p}(x) = \det(L(x))$ . Set  $f(x) := -\ln \check{p}(x)$  and let  $f^{(k)}(x)[h] := \frac{d^k f(x+th)}{dt^k} \Big|_{t=0}$  be the  $k^{\text{th}}$  directional derivative of  $f$  in direction  $h$ . Then for each even integer  $k$

$$f^{(k)}(x)[h] \geq 0$$

for all  $x$  in  $S$  the component of  $\{x : \check{p}(x) > 0\}$  containing 0 and all  $h \in \mathbb{R}^n$ .

*Proof.* Let  $a(x) = L(x)^{-1}$ . First, note the standard fact

$$\frac{df(x+th)}{dt} = -\text{trace}(a(x+th)\tilde{L}(h))$$

where  $\tilde{L}$  is the purely linear part of  $L(h)$ . Since the trace functional is linear, we have

$$\frac{d^2f(x+th)}{dt^2} = \text{trace}(a(x+th)\tilde{L}(h)a(x+th)\tilde{L}(h)) = \text{trace}(b(t)^2)$$

where  $b(t) = a(x+th)\tilde{L}(h)$ . Using the formula

$$\frac{db(t)}{dt} = -a(x+th)\tilde{L}(h)a(x+th)\tilde{L}(h) = -b(t)^2,$$

we can inductively show that

$$\frac{d^k f(x+th)}{dt^k} = (-1)^k \cdot 2 \cdot 3 \cdots (k-1) \text{trace}(b(t)^k).$$

So we have the formula

$$f^{(k)}(x)[h] = (-1)^k (k-1)! \text{trace}((a(x)\tilde{L}(h))^k).$$

Set  $B := a(x)\tilde{L}(h)$  and write  $f^{(k)}(x)[h]$  as

$$(-1)^k (k-1)! \text{trace}\left(a(x)^{\frac{1}{2}} (B^T)^{\frac{(k-2)}{2}} \tilde{L}(h) a(x) \tilde{L}(h) \left(B^{\frac{(k-2)}{2}}\right) a(x)^{\frac{1}{2}}\right)$$

when  $k$  is even. The above is nonnegative if  $a(x) = L(x)^{-1} > 0$ .  $\square$

### 4.3 SDP Representation

As we saw in the preceding section not every convex semialgebraic set  $S$  has an LMI representation. A set  $S \subset \mathbb{R}^n$  lifts to a set  $\hat{S} \subset \mathbb{R}^{n+N}$  means that  $S$  is the image of  $\hat{S}$  under the coordinate projection of  $\mathbb{R}^{n+N}$  into  $\mathbb{R}^n$ . A set  $S$  is semidefinite representable or SDP representable if there exist an  $N$  and a LMI representable set  $\hat{S} \subset \mathbb{R}^{n+N}$  such that  $S$  lifts to  $\hat{S}$ . If so, we say  $S$  has an SDr or is SDr. The original book by Nesterov and Nemirovski [17] gave collections of examples of SDr sets, thereby leading to the question: which sets are SDP representable? In Sect. 4.3.1 of his excellent 2006 survey [18] Nemirovski commented “this question seems to be completely open”.

Recall an example. The convex set in Example 4.1 is

$$T := \{x \in \mathbb{R}^2 : 1 - (x_1^4 + x_2^4) \geq 0\}$$

does not have an LMI representation [11], since it is not rigidly convex. However, the set  $T$  is the projection into  $x$ -space of the SDr set

$$\hat{S} := \left\{ (x, w) \in \mathbb{R}^2 \times \mathbb{R}^2 : \begin{bmatrix} 1 + w_1 & w_2 \\ w_2 & 1 - w_1 \end{bmatrix} \succeq 0, \begin{bmatrix} 1 & x_1 \\ x_1 & w_1 \end{bmatrix} \succeq 0, \begin{bmatrix} 1 & x_2 \\ x_2 & w_2 \end{bmatrix} \succeq 0 \right\}.$$

Suppose  $S$  is a basic closed semialgebraic set of the form

$$S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\} \quad (4.8)$$

defined by polynomials  $g_i(x)$ . All closed semialgebraic sets are finite unions of such sets. Since a compact LMI representable set is both convex and semialgebraic, and the projection of a semialgebraic set is also semialgebraic (by the Tarski–Seidenberg principle), an SDr set must also be convex and semialgebraic. *What conditions guarantee that  $S$  is SDP representable?*

Recently, there has been substantial progresses on this problem which we sketch here. Lasserre [13] and Parrilo [21] independently proposed a natural construction of SDP lifts using moments and sum of squares techniques with the aim of producing SDr. Parrilo [21] proved the construction gives an SDr in the two dimensional case when the boundary of  $S$  is a single rational planar curve of genus zero. Lasserre [13] showed the construction can give arbitrarily accurate approximations to compact  $S$ , and it gives SDr if *almost all* positive affine functions on  $S$  have SOS type representations with uniformly bounded degrees. In this section we state sufficient conditions for an SDr to exist, and our sufficient conditions are not far away from being necessary; this follows [7, 8]. Later, in Sect. 4.5 we give the more concrete “SOS concave condition”, and in Sect. 4.6 we give strict concavity or quasi-concavity type conditions. All of this is an account of results in [7–9, 13].

### 4.3.1 The Main Result for SDP Representability

If  $S$  is a semialgebraic set then write the Zariski closure of its topological boundary  $\partial S$  as a union like

$$Z_1 \cup \dots \cup Z_m$$

with each  $Z_j$  being irreducible varieties. This varietal decomposition may not be unique and we call one with the minimal number of  $Z_j$  minimal of  $\partial S$ .

We say  $Z_i$  is nonsingular at  $x \in Z_i$  if it has a defining polynomial  $Z_i = \{x \in \mathbb{R}^n : g_i(x) = 0\}$  with  $\nabla g_i(x) \neq 0$ . If in addition

$$-v^T \nabla^2 g_i(x)v > 0, \quad \forall v \in \nabla g_i(x)^\perp, v \neq 0 \quad (4.9)$$

in other words, the Hessian of  $g_i$  compressed to the tangent space (the second fundamental form) is negative definite, we say that  $Z_i$  has positive curvature at  $x$ . A standard fact in geometry (follows from the chain rule) is that these properties do not depend on the choice of the defining function. Issues of orientation and sign of the curvature may be a little disorienting to the reader who is referred to Appendix 4.7.2.

Intuitively, a variety  $Z$  is nonsingular at a point  $u$  means it does not cross itself at  $u$ . A variety  $Z_i$  crossing a different  $Z_j$  does not constitute a singularity, and in our situation this causes no difficulties, while a variety crossing itself does. As to positive curvature, note that the boundary of any convex set on its smooth parts has nonnegative curvature.

Theorem 3.9 of [8] implies the following.

**Theorem 4.6.** *Suppose  $S$  is a convex compact basic<sup>2</sup> semialgebraic set with nonempty interior. Let  $\mathcal{V}$  be a varietal decomposition of  $\partial S$ . For each  $u \in \partial S$ , if each  $Z_j$  in  $\mathcal{V}$  containing  $u$*

1. *Is nonsingular at  $u$*
2. *Has positive curvature at  $u$*

*then  $S$  is SDP representable.*

*Proof.* We shall give a rough idea of how this is proved later in the chapter. There is a “construction” involving moment matrix relaxations which “produces” an SDR of the set  $S$ .  $\square$

Conversely, if  $S$  is convex with nonsingular defining functions, then its boundary has nonnegative curvature. Thus the positive curvature assumption is not a huge restriction beyond being convex. The nonsingularity assumption is possibly more nettlesome.

Many variations of Theorem 4.6 are given in [8]. For example, the polynomial  $g_i$  is called sos-concave if  $-\nabla^2 g_i(x) = W(x)^T W(x)$  for some possibly nonsquare matrix polynomial  $W(x)$ . Then we can replace nonsingularity and the curvature hypotheses on  $Z_i \cap \partial S$  by the condition that  $g_i$  is sos-concave, while we still have  $S$  is SDR. Sos-concavity will be discussed more in Sect. 4.5.

### 4.3.2 Convex Hulls of Semialgebraic Sets

In this section, we consider the problem of finding the convex hull of a nonconvex semialgebraic set  $T$ . Without loss of generality, assume  $T$  is closed, otherwise consider its closure. No matter whether  $T$  is convex or not, its convex hull  $\text{conv}(T)$  is always convex and semialgebraic (Theorem 2.2.1 in [3]). By Theorem 2.7.2 in [3],

---

<sup>2</sup>The result does not require a basic semialgebraic convex set, a semialgebraic convex set will do, although description of the boundary varieties is a bit more complicated. See Theorem 4.8 and subsequent remarks.

the closure of  $\text{conv}(T)$  is a union of basic closed semialgebraic sets. A fundamental problem in convex geometry and semidefinite programming is to find an SDR of  $\text{conv}(T)$ . This section will address this problem and state sufficient conditions and necessary conditions for the SDP representability of  $\text{conv}(T)$ .

A general property of SDP representations and convex hulls is:

**Theorem 4.7 (Theorem 4.2, [8]).** *Let  $T_1, \dots, T_m$  be bounded semialgebraic sets. If each  $\text{conv}(T_k)$  is SDP representable, then the convex hull of  $\cup_{k=1}^m T_k$  is also SDP representable.*

The proof of this amounts to a formula which will be given later in Theorem 4.15.

Now we turn to more refined results. Let  $T$  be a compact nonconvex set with boundary  $\partial T$ . Obviously  $\text{conv}(T)$  is the convex hull of the boundary  $\partial T$ . Some part of  $\partial T$  might be in the interior of  $\text{conv}(T)$  and will not contribute to  $\text{conv}(T)$ . So we are motivated to define the convex boundary  $\partial_c T$  of  $T$  as

$$\partial_c T = \left\{ u \in T : \ell^T u = \min_{x \in T} \ell^T x \text{ for some } \ell \in \mathbb{R}^n \text{ with } \|\ell\| = 1 \right\} \subseteq \partial T. \quad (4.10)$$

Geometrically,  $\partial_c T$  is the set of extremal points of  $\text{conv}(T)$ .

**Proposition 4.3.** *If  $T$  is compact, then  $\text{conv}(\partial_c T) = \text{conv}(T)$  and  $\partial_c T$  is also compact.*

*Proof.* Obviously  $\text{conv}(\partial_c(T)) \subseteq \text{conv}(T)$ . We need to prove  $\text{conv}(\partial_c(T)) \supseteq \text{conv}(T)$ . It suffices to show that if  $u \notin \text{conv}(\partial_c T)$  then  $u \notin \text{conv}(T)$ . For any  $u \notin \text{conv}(\partial_c T)$ , by the Convex Set Separation Theorem, there is a vector  $\ell$  of unit length and a positive number  $\delta > 0$  such that

$$\ell^T u < \ell^T x - \delta, \quad \forall x \in \text{conv}(\partial_c T).$$

Let  $v \in T$  minimize  $\ell^T x$  over  $T$ , which must exist due to the compactness of  $T$ . Then  $v \in \partial_c T$  and hence

$$\ell^T u < \ell^T v - \delta = \min_{x \in T} \ell^T x - \delta.$$

Therefore,  $u \notin \text{conv}(T)$ .

Clearly  $\partial_c T$  is bounded and closed by its definition. So  $\partial_c T$  is compact.  $\square$

If  $T$  is not compact, then Proposition 4.3 might not be true. For instance, for set  $T = \{x \in \mathbb{R}^2 : \|x\|^2 \geq 1\}$ , the convex boundary  $\partial_c T = \emptyset$ , but  $\text{conv}(T)$  is the whole space. When  $T$  is not compact, even if  $\text{conv}(\partial T) = \text{conv}(T)$ , it is still possible that  $\text{conv}(\partial_c T) \neq \text{conv}(T)$ . As a counterexample, consider the set

$$W = \{(0,0)\} \cup \{x \in \mathbb{R}_+^2 : x_1 x_2 \geq 1\}.$$

It can be verified that  $\text{conv}(W) = \text{conv}(\partial W)$ ,  $\partial_c W = \{(0,0)\}$  and  $\text{conv}(\partial_c W) \neq \text{conv}(W)$ .

The next theorem (following from Theorem 4.5 [8]) concerns the SDP representability of convex hulls of semialgebraic sets.

**Theorem 4.8 (Sufficient conditions for SDP representability of convex hulls).** *Assume  $T = \bigcup_{k=1}^L T_k$  is a compact semialgebraic set with each  $T_k$  being a basic semialgebraic set, whose boundary has varietal decomposition denoted  $\mathcal{V}(k)$ . If for each  $u \in \partial_c T$  and  $Z_{k_i}$  in  $\mathcal{V}(k)$  with  $u$  in  $Z_{k_i}$ ,*

1.  $Z_{k_i}$  is non singular at  $u$ .
2.  $Z_{k_i}$  has positive curvature at  $u$ .
3.  $T_k$  has interior near  $u$ , that is, for any  $\varepsilon > 0$ , there exists an interior point of  $T_k$  which lies in the ball  $B(u, \varepsilon)$ .

then  $\text{conv}(T)$  is SDP representable.

The proof is based on a very powerful technique of localization. We shall consider sets of the form  $W_k(u) := T_k \cap \bar{B}(u, \varepsilon)$  where  $\bar{B}(u, \varepsilon)$  denotes the closed ball with center  $u$  and radius  $\varepsilon$ .

*Proof.* Suppose  $u$  in the  $\partial_c T \cap Z_{k_i}$  as in the hypothesis of the theorem. Positive curvature (and nonsingularity) imply there is a small enough  $\varepsilon$  such that each  $W_k(u) := T_k \cap \bar{B}(u, \varepsilon)$  is convex. The boundary  $\partial W_k(u)$  is nonsingular and has positive curvature. The 3rd condition implies  $W_k(u)$  has interior. Thus  $W_k(u)$  satisfies the hypothesis of Theorem 4.6, so it has an SDr.

Select a finite collection  $\mathcal{W}$  of the  $W_k(u)$  which is a cover of  $\partial T_c$ . Theorem 4.7 applied to the  $W_k(u)$  in  $\mathcal{W}$  produces an SDr for the convex hull of  $\mathcal{W}$ . Proposition 4.3 implies that the convex hull of  $\mathcal{W}$  is  $\text{conv}(\bigcup_{k=1}^L T_k)$ .  $\square$

Clearly Theorem 4.8 contains Theorem 4.6 as a special case. However, one can extend Theorem 4.6 to hold for any compact convex semialgebraic set with nonempty interior. Use the fact that every semialgebraic set is a finite union of basic semialgebraic sets (Proposition 2.1.8 in [3]), we can break it into basic semialgebraic sets  $T = \bigcup_{k=1}^L T_k$  and then use the localization technique as in the proof of Theorem 4.8.

### 4.3.3 Necessary Conditions for SDP Representability

Our sufficient conditions in Theorem 4.8 are close to being necessary, because of the following fact about “pieces of  $\partial S$ ”. If  $Z_i$  is an irreducible component of the Zariski closure of  $\partial S$  and  $Z_i \cap B(u, \varepsilon) = \partial S \cap B(u, \varepsilon)$  for some  $u \in \partial S$  and  $\varepsilon > 0$ , then  $Z_i$  has nonnegative curvature at  $u$ . This is summarized in as follows, which is basically implied by Theorem 3.8 of [8].

**Theorem 4.9 (Necessary conditions for SDP representation).** *Suppose  $S$  is a compact set that is SDP representable. Then*

1.  *$S$  is a finite union of basic closed semialgebraic sets.*
2. *Each piece  $Z_i \cap B(u, \varepsilon)$  of the boundary of  $S$  has nonnegative curvature on its smooth points.*

## 4.4 Moment Type Constructions

The moment type constructions of SDP representations for convex sets were proposed by Lasserre [13] and independently by Parrilo in the two dimensional case, like in [21].

In this section we describe the construction with less concern as to when it works. In the subsequent sections we prove various sufficient conditions (like sos-convexity, strict convexity or quasi-convexity) guaranteeing the moment type constructions give correct SDr.

### 4.4.1 The Conceptual Idea

We begin presentation of the construction with a motivating idea. Consider a compact convex set  $S \subset \mathbb{R}^n$ . Let  $\mathcal{M}$  denote the space of Borel measures on  $S$  and let  $\hat{\mathcal{M}}$  denote the convex subset of all nonnegative mass one measures. The Krein Millman Theorem [5] says that  $\hat{\mathcal{M}}$  projects down onto  $S$  via

$$P(\mu) := \int_S x d\mu(x) \quad \mu \in \hat{\mathcal{M}}.$$

Unfortunately,  $\hat{\mathcal{M}}$  is infinite dimensional, so it is unsuitable for SDP representation. The Lasserre and Parrilo type constructions, which will be sketched later, are to cut down  $\hat{\mathcal{M}}$  by looking at it as the set of all positive mass one linear functionals on the polynomials of some fixed degree  $2N$ . Moment and sum of squares (SOS) techniques show that this gives an LMI, denoted by  $\mathcal{L}_N$ , for each degree  $N$ , and that the projection into  $x$ -space of the set  $\hat{\mathcal{M}}_N := \{(x, y) : \mathcal{L}_N(x, y) \geq 0\}$  contains  $S$  for all  $N$ . The open question remaining is whether there exists an integer  $N$  large enough to produce the equality.

The validity of this general type of construction has been supported by very nice recent findings on the SDP representation of convex sets. Parrilo [21] proved this gives a correct SDP representation in the two dimensional case when the

boundary of  $S$  is a single rational planar curve of genus zero. Lasserre [13] proved this construction can give arbitrarily accurate approximations for compact convex sets when  $N$  goes to infinity. He also gave nice sufficient algebraic conditions like PP-BDR or S-BDR to be defined later.

#### 4.4.2 A Basic Moment Relaxation

Lasserre [13] and Parrilo [21] proposed recipes for a natural SDP representation via moments.

Suppose  $S$  is a convex set given in the form

$$S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}. \quad (4.11)$$

Here every  $g_i(x)$  is a polynomial. Let

$$d = \max\{\lceil \deg(g_1)/2 \rceil, \dots, \lceil \deg(g_m)/2 \rceil\}. \quad (4.12)$$

Write every  $g_i$  as

$$g_i(x) = \sum_{|\alpha| \leq 2d} g_\alpha^{(i)} x^\alpha.$$

If we let  $y_\alpha = x^\alpha$  for every  $\alpha$ , then  $S$  is equivalent to

$$\left\{ x \in \mathbb{R}^n : L_{g_1}(y) \geq 0, \dots, L_{g_m}(y) \geq 0, \text{ every } y_\alpha = x^\alpha \right\}.$$

In the above, the linear functional  $L_{g_i}(y)$  is defined as

$$L_{g_i}(y) = \sum_{|\alpha| \leq 2d} g_\alpha^{(i)} y_\alpha.$$

Here  $y$  is a moment vector indexed by vectors  $\alpha \in \mathbb{N}^n$  with  $|\alpha| \leq 2d$ . Note the trivial matrix inequality

$$[x]_d [x]_d^T \geq 0,$$

where  $[x]_d$  denotes the column vector of monomials of degrees  $\leq d$ , i.e.,

$$[x]_d = \begin{bmatrix} 1 & x_1 & \cdots & x_1^2 & x_1 x_2 & \cdots & x_n^d \end{bmatrix}^T.$$

Now we define the linear matrix pencil  $M_d(y)$  such that

$$M_d(y) = [x]_d [x]_d^T \quad \text{whenever every } y_\alpha = x^\alpha. \quad (4.13)$$

The  $M_d(y)$  is called the moment matrix of order  $d$  in  $n$  variables. Thus,  $S$  can be equivalently described as

$$S = \left\{ x \in \mathbb{R}^n : \begin{array}{l} L_{g_1}(y) \geq 0, \dots, L_{g_m}(y) \geq 0, \\ M_d(y) \geq 0, \\ \text{every } y_\alpha = x^\alpha \end{array} \right\}.$$

In the above, every relation is linear except  $y_\alpha = x^\alpha$  for  $|\alpha| > 1$ . Note  $y_0 = x^0 = 1$ . If we remove these nonlinear constraints,  $S$  is clearly contained in the SDr set

$$R = \left\{ x \in \mathbb{R}^n : \begin{array}{l} L_{g_1}(y) \geq 0, \dots, L_{g_m}(y) \geq 0, \\ y_0 = 1, \quad M_d(y) \geq 0, \\ x_1 = y_{e_1}, \dots, x_n = y_{e_n} \end{array} \right\}. \quad (4.14)$$

Here  $e_i$  denotes the  $i$ -th standard unit basis vector of  $\mathbb{R}^n$ . Note that  $R$  is the projection into  $x$ -space of a higher dimensional LMIr set. The integer  $d$  in (4.14) is defined in (4.12). Since  $S \subset R$ , we also say  $R$  is a relaxation of  $S$ .

We now illustrate above constructions with two examples.

*Example 4.4.* Consider the set  $S = \{x \in \mathbb{R}^n : g(x) \geq 0\}$  where

$$g(x) = 1 - (x_1^4 + x_2^4 - x_1^2 x_2^2).$$

From the previous construction, we know this  $S$  is contained in the set  $R$  defined by

$$\begin{aligned} 1 - y_{40} - y_{04} + y_{22} &\geq 0, \quad y_0 = 1, \\ \begin{bmatrix} 1 & x_1 & x_2 & y_{20} & y_{11} & y_{02} \\ x_1 & y_{20} & y_{11} & y_{30} & y_{21} & y_{12} \\ x_2 & y_{11} & y_{03} & y_{21} & y_{12} & y_{03} \\ y_{20} & y_{30} & y_{21} & y_{40} & y_{31} & y_{22} \\ y_{11} & y_{21} & y_{12} & y_{31} & y_{22} & y_{13} \\ y_{02} & y_{12} & y_{03} & y_{22} & y_{13} & y_{04} \end{bmatrix} &\geq 0. \end{aligned}$$

The matrix above is the second order moment matrix. There are 12 auxiliary variables  $y_{ij}$ . Later, we will see that  $R = S$  for this example.

*Example 4.5.* Consider the set  $S = \{x \in \mathbb{R}^n : 1 - p(x) \geq 0\}$  where

$$p(x) = (x^d)^T B x^d.$$

Here  $d > 0$  is an integer and  $B = B^T \succeq 0$ , and  $x^d$  denotes

$$x^d = [x_1^d \ x_2^d \ \cdots \ x_n^d]^T.$$

From the previous construction,  $S$  is contained in  $R$  defined as

$$R = \left\{ x : \begin{array}{l} \exists y_\alpha (\alpha \in \mathbb{N}^n), 1 - \sum_{i,j=1}^n B_{ij} y_{d(e_i+e_j)} \geq 0, \\ y_0 = 1, M_d(y) \geq 0, \\ x_1 = y_{e_1}, \dots, x_n = y_{e_n} \end{array} \right\}.$$

Actually, we will see later that  $S = R$  for this instance.

#### 4.4.3 Refined Moment Type Relaxations

Tighter moment type relaxations would be obtained if we go further by using higher order moments. We here describe two types of moment constructions: Putinar and Schmüdgen type SDP lifting.

For the purpose of refined constructions, we need define localizing moment matrix. Let  $p$  be a polynomial and  $2N \geq \deg(p)$ . Write

$$p(x)[x]_{N-k}[x]_{N-k}^T = \sum_{0 \leq |\alpha| \leq 2N} A_\alpha^{(N)} x^\alpha, \quad k = \lceil \deg(p)/2 \rceil,$$

in order to define a linear pencil  $L_p^{(N)}$  by

$$L_p^{(N)}(y) = \sum_{0 \leq |\alpha| \leq 2N} A_\alpha^{(N)} y_\alpha.$$

Clearly, if  $p(x)$  is nonnegative on  $S$ , then all  $x \in S$  satisfy

$$L_p^{(N)}(y) \geq 0, \quad \text{if each } y_\alpha = x^\alpha.$$

Let  $g_0(x) \equiv 1$ , then  $L_{g_0}^{(d)} = M_d(y)$  in (4.13). Clearly, all  $g_0(x), g_1(x), \dots, g_m(x)$  are nonnegative on  $S$ . Hence all  $x \in S$  belong to the set

$$\tilde{S}_N = \left\{ x \in \mathbb{R}^n : \begin{array}{l} L_{g_i}^{(N)}(y) \geq 0, i = 0, 1, \dots, m \\ y_0 = 1, x_1 = y_{e_1}, \dots, x_n = y_{e_n} \end{array} \right\}. \quad (4.15)$$

That is,  $\tilde{S}_N$  is a relaxation of  $S$ , and is the projection of a higher dimensional LMIr set into the  $x$ -space. When  $S = \tilde{S}_N$ , we say (4.15) is a Putinar type SDr of  $S$ .

Obviously, the product of any subset of  $g_1(x), \dots, g_m(x)$  is nonnegative on  $S$ . For every  $v \in \{0, 1\}^m$ , define  $g_v(x) := g_1^{v_1}(x) \cdots g_m^{v_m}(x)$  (note  $g_0(x) \equiv 1$ ). Note that each  $g_v(x)$  is nonnegative on  $S$ . So all  $x \in S$  satisfy

$$y_0 = 1, \quad L_{g_v}^{(N)}(y) \geq 0, \quad \forall v \in \{0, 1\}^m$$

if each  $y_\alpha = x^\alpha$ . So,  $S$  is contained in the set

$$\hat{S}_N = \left\{ x \in \mathbb{R}^n : \begin{array}{l} L_{g_\nu}^{(N)}(y) \geq 0, \quad \forall \nu \in \{0, 1\}^m, \\ y_0 = 1, x_1 = y_{e_1}, \dots, x_n = y_{e_n} \end{array} \right\}. \quad (4.16)$$

Thus,  $\hat{S}_N$  is also a relaxation of  $S$ , and is the projection of a higher dimensional LMIR set into the  $x$ -space. When  $S = \hat{S}_N$ , we say (4.16) is a Schmüdgen type SDr of  $S$ . Clearly,  $\hat{S}_N \subset \tilde{S}_N$  because (4.16) has extra LMIs in addition to those in (4.15).

We will see in later sections that both  $\hat{S}_N$  and  $\tilde{S}_N$  are equal to  $S$  for  $N$  big enough, under conditions like strict concavity or strict quasi-concavity on the defining polynomials  $g_1, \dots, g_m$ .

## 4.5 Sos-Convex Sets

In this section, we prove a sufficient condition justifying the basic moment construction (4.14) is an SDP representation for the set  $S$  defined by (4.11). This is the sos-convexity condition.

Recall that a polynomial  $p$  is said to be a sum of squares (SOS) if  $p(x) = w(x)^T w(x)$  for some column vector polynomial  $w$ . A necessary condition for  $p$  to be SOS is that  $p(x)$  is nonnegative in the whole space  $\mathbb{R}^n$ , but the reverse might not be true. We refer to [26] for a survey on nonnegative and SOS polynomials. A symmetric matrix polynomial  $P \in \mathbb{R}[x]^{n \times n}$  is SOS if there exists a possibly nonsquare matrix polynomial  $W$  with  $n$  columns such that  $P(x) = W(x)^T W(x)$ . The defining polynomial  $g_i$  is called sos-concave if the negative Hessian  $-\nabla^2 g_i$  is SOS. Similarly,  $g_i$  is called sos-convex if the Hessian  $\nabla^2 g_i$  is SOS. If every  $g_i$  is sos-concave, we say the set  $S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}$  is sos-convex.

The main result of this section is:

**Theorem 4.10.** *Assume  $S$  has nonempty interior. If every  $g_i(x)$  is sos-concave with degree at most  $2d$ , then (4.14) is an SDP representation for  $S$ .*

Now we are going to present the proof. It forms a basis for most of our theorems proving that the finite moment type constructions produce SDP representations. The standard approach for justifying lifted LMIs is to applying separating linear functionals. For each vector  $\ell \in \mathbb{R}^n$ , let  $\ell^*$  be the minimum value of  $\ell^T x$  over the set  $S$ ,  $u \in S$  be the minimizer, which must lie on the boundary  $\partial S$ . If  $S$  has nonempty interior (the Slater's condition holds) and every  $g_i$  is concave, then there exist Lagrange Karush–Kuhn–Tucker multipliers  $\lambda_1 \geq 0, \dots, \lambda_m \geq 0$  such that

$$\ell = \lambda_1 \nabla g_1(u) + \dots + \lambda_m \nabla g_m(u).$$

Clearly, the Lagrangian

$$f_\ell(x) := \ell^T x - \ell^* - \lambda_1 g_1(x) - \cdots - \lambda_m g_m(x) \quad (4.17)$$

is convex, nonnegative everywhere, vanishes at  $u$  as well as its gradient  $\nabla f_\ell(u) = 0$  (see Lasserre [13]). Under the assumption that  $f_\ell(x)$  is SOS for every  $\ell \in \mathbb{R}^n$ , Lasserre [13] showed (4.14) is an SDP representation of  $S$ . If  $f_\ell$  is not SOS for some  $\ell \in \mathbb{R}^n$ , then (4.14) might no longer represent  $S$ . To justify that (4.14) is a correct lifted LMI for  $S$ , we need to prove that every  $f_\ell(x)$  is SOS. This is the case when  $S$  is sos-convex.

The key parts of the proof for Theorem 4.10 are the following lemmas, which were given in [7].

**Lemma 4.1.** *If a symmetric matrix polynomial  $P \in \mathbb{R}[x]^{r \times r}$  is SOS, then for any  $u \in \mathbb{R}^n$ , the symmetric matrix polynomial*

$$\int_0^1 \int_0^t P(u + s(x-u)) ds dt$$

is SOS. In particular, when  $r = 1$ , the above integral is an SOS polynomial.

*Proof.* Let  $2k = \deg(P)$ . Then  $P$  being SOS implies

$$\xi^T P(x) \xi = [\xi x^k]^T A^T A [\xi x^k]$$

for matrix  $A$ . Here  $[\xi x^k]$  denotes the vector of monomials

$$\begin{bmatrix} \xi_1 [x^k]^T & \cdots & \xi_r [x^k]^T \end{bmatrix}^T.$$

Note that  $\xi^T P(x) \xi$  has degree 2 in  $\xi$ . If  $P(x)$  is SOS, we can assume the above  $A$  exists. In monomial vector  $[\xi x^k]$  we replace  $x$  by  $u + s(x-u)$ . Each entry of  $[\xi(u + s(x-d))^k]$  is a polynomial in  $x$  whose coefficients are polynomials in  $u$  and  $s$ . So there exists a matrix polynomial  $C(u, s)$  such that  $[\xi(u + s(x-d))^k] = C(u, s)[\xi x^k]$ . Therefore we have

$$\int_0^1 \int_0^t \xi^T P(u + s(x-u)) \xi ds dt = [\xi x^k]^T B^T B [\xi x^k]$$

where  $B$  is a matrix such that

$$\int_0^1 \int_0^t C(u, s)^T A^T A C(u, s) ds dt = B^T B.$$

Therefore, the double integral of matrix polynomial in the lemma is SOS.  $\square$

**Lemma 4.2.** Let  $p$  be polynomial such that  $p(u) = 0$  and  $\nabla p(u) = 0$  for some point  $u \in \mathbb{R}^n$ . If the Hessian  $\nabla^2 p$  is SOS, then  $p$  is SOS.

*Proof.* Let  $q(t) = p(u + t(x - u))$  be a univariate polynomial in  $t$ . Then

$$q''(t) = (x - u)^T \nabla^2 p(u + t(x - u))(x - u).$$

So we have

$$p(x) = q(1) = (x - u)^T \left( \int_0^1 \int_0^t \nabla^2 p(u + s(x - u)) ds dt \right) (x - u).$$

Since  $\nabla^2 p(x)$  is SOS, the middle double integral above should also be SOS, by Lemma 4.1. So  $p(x)$  is also SOS.  $\square$

Now we are able to present the proof of Theorem 4.10.

*Proof.* It is obvious that  $S$  is contained in the set  $R$  defined in (4.14). If they are not equal, there must exist some  $\hat{y} \in R$  such that  $\hat{x} = (\hat{y}_{e_1}, \dots, \hat{y}_{e_n}) \notin S$ . Since  $S$  is closed, there exists a supporting hyperplane of  $S$  that excludes  $\hat{x}$ , i.e., there exists a unit length vector  $\ell \in \mathbb{R}^n$  such that

$$\begin{aligned} \ell^T x - \ell^* &\geq 0, \quad \forall x \in S, \quad \ell^T \hat{x} - \ell^* < 0, \\ \ell^T u - \ell^* &= 0, \quad \exists u \in \partial S. \end{aligned}$$

Then  $u$  is a minimizer for the above. Since  $\text{int}(S) \neq \emptyset$ , we have seen earlier that there must exist  $\lambda_1 \geq 0, \dots, \lambda_m \geq 0$  such that the Lagrangian  $f_\ell(x)$  in (4.17) is a convex nonnegative polynomial satisfying  $f_\ell(u) = 0$  and  $\nabla f_\ell(u) = 0$ . Since

$$\nabla^2 f_\ell(x) = \sum_{i=1}^m \lambda_i (-\nabla^2 g_i(x)),$$

we know  $f_\ell(x)$  is sos-convex. Then, Lemma 4.2 implies  $f_\ell(x)$  is SOS. Since the degree of  $f_\ell$  is at most  $2d$ , there exists a symmetric matrix  $W \succeq 0$  such that the identity

$$\ell^T x - \ell^* = \sum_{i=1}^m \lambda_i g_i(x) + [x]_d^T W [x]_d$$

holds. In the above identity, replace each monomial  $x^\alpha$  by  $\hat{y}_\alpha$ , then we get

$$\ell^T \hat{x} - \ell^* = \sum_{i=1}^m \lambda_i L_{g_i}(\hat{y}) + \text{Trace}(W \cdot M_d(\hat{y})) \geq 0,$$

which contradicts the previous assertion  $\ell^T \hat{x} - \ell^* < 0$ .  $\square$

We would like to remark that not every concave polynomial is sos-concave, as found by Ahmadi and Parrilo [1]. Typically sos-concavity is a very strong condition. Actually, a nonnegative convex polynomial is usually not SOS, as proved by Blekherman [2]. So generally (4.14) does not represent  $S$  when the polynomials  $g_i$  are just concave over  $\mathbb{R}^n$ . On the other hand, it is tractable to check sos-concavity. Obviously,  $-\nabla^2 g_i(x)$  is SOS if and only if the polynomial

$$-\sum_{k,\ell=1}^n \frac{\partial^2 g_i(x)}{\partial x_k \partial x_\ell} z_k z_\ell$$

in  $(x, z)$  is SOS. This can be checked numerically by solving a single SDP feasibility problem.

*Example 4.6 (Revisiting Example 4.4).* The polynomial  $g(x)$  is sos-concave, because

$$-\nabla^2 g(x) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \underbrace{\begin{bmatrix} 12 & -4 \\ -4 & 12 \end{bmatrix}}_{\geq 0} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

By Theorem 4.10, we know the set  $R$  there is an SDr for  $S$ .

*Example 4.7 (Revisiting Example 4.5).* The set  $S$  is sos-convex because

$$\nabla^2 p(x) = \text{diag}(x^{d-1}) \cdot W \cdot \text{diag}(x^{d-1})$$

where the symmetric matrix  $W$  is given as

$$W = d^2 B + (3d^2 - 2d)\text{diag}(B).$$

When  $B \geq 0$  and  $d \geq 1$ , we have  $W \geq 0$ . So  $p(x)$  is sos-convex. By Theorem 4.10, we know  $R$  is an SDr for  $S$ .

## 4.6 Strictly Convex Sets

We advance through this section and the next along weaker and weaker hypotheses with each yielding valid SDP representations.

### 4.6.1 Strictly Concave Defining Polynomials

Recall the Putinar type SDP relaxation  $\tilde{S}_N$  in (4.15). We have already seen that  $S \subset \tilde{S}_N$  for each  $2N \geq \max_i \deg(g_i)$ . Since  $L_{g_i}^{(N)}$  is a leading principle submatrix of  $L_{g_i}^{(N+1)}$ ,

we can see that  $\tilde{S}_{N+1}$  is contained in  $\tilde{S}_N$ . Thus, we get the nesting containment relation

$$\tilde{S}_N \supseteq \tilde{S}_{N+1} \supseteq \cdots \supseteq S.$$

A natural question is whether there exists a finite integer  $N$  such that  $\tilde{S}_N = S$ .

One typical approach to this question is to apply separating linear functionals if  $\hat{S}_N$  is different from  $S$ , as we did in the preceding section. Suppose  $\ell^T x$  is a linear functional with minimum  $\ell^*$  on  $S$  and a minimizer  $u \in \partial S$ , which must be true if  $S$  is compact. When  $\text{int}(S) \neq \emptyset$  and every  $g_i$  is a concave nonconstant polynomial on  $S$ , we must have some  $u \in \text{int}(S)$  satisfying  $g_i(u) > 0$  for every  $i$ , because otherwise at least one  $g_i$  vanishes identically in  $\text{int}(S)$  and it should be the zero polynomial. So the Slater's condition holds, there exist Lagrange multipliers  $\lambda_1 \geq 0, \dots, \lambda_m \geq 0$  such that  $\ell = \sum_i \lambda_i \nabla g_i(u)$  and the Lagrangian

$$f_\ell(x) = \ell^T x - \ell^* - \sum_i \lambda_i g_i(x)$$

is nonnegative on  $S$ , because

$$f_\ell(x) \geq f_\ell(u) + \nabla f_\ell(u)^T (x - u) = 0, \quad \forall x \in S.$$

So, we could wish  $f_\ell(x)$  to have Putinar type representation

$$f_\ell(x) = \sigma_0 + \sigma_1 g_1 + \cdots + \sigma_m g_m$$

for certain SOS polynomials  $\sigma_i(x)$ . Note that this representation is not implied by Putinar's Positivstellensatz [23] because  $f_\ell(x)$  vanishes at  $u$  on  $S$ . In addition, to get a Putinar type representation, one needs the archimedean condition (AC): there exist SOS polynomials  $s_0(x), s_1(x), \dots, s_m(x)$  and a number  $M > 0$  big enough such that

$$M - \sum_{i=1}^n x_i^2 = s_0(x) + s_1(x)g_1(x) + \cdots + s_m(x)g_m(x).$$

Clearly, AC implies  $S$  is compact, but the reverse is not always true. However, a compact  $S$  can be strengthened to satisfy AC by adding a “redundant” constraint like  $M - \sum_{i=1}^n x_i^2 \geq 0$  for sufficiently large  $M$ .

To validate  $\tilde{S}_N = S$  for some finite integer  $N$  amounts to proving that for all  $\ell$  the Lagrangian  $f_\ell(x)$  has Putinar type representation with uniform (in  $\ell$ ) degree bounds on SOS polynomials  $\sigma_i(x)$ . This is equivalent to proving a property of  $S$  called Putinar–Prestel’s Bounded Degree Representation (PP-BDR) of order  $N$  (see Lasserre [13]), that is, for *almost every*  $(a, b) \in \mathbb{R}^n \times \mathbb{R}$

$$a^T x + b > 0 \text{ on } S \quad \Rightarrow \quad a^T x + b = \varphi_0(x) + \varphi_1(x)g_1(x) + \cdots + \varphi_m(x)g_m(x)$$

for some SOS polynomials  $\varphi_i(x)$  with degree bounds  $\deg(\varphi_i g_i) \leq 2N$ .

Similarly, for Schmüdgen type SDP relaxation  $\hat{S}_N$  in (4.16), we also have the nesting containment relation

$$\hat{S}_N \supseteq \hat{S}_{N+1} \supseteq \cdots \supseteq S.$$

To justify  $\hat{S}_N = S$  for some finite integer  $N$ , we need to prove  $S$  has the so-called Schmüdgen's Bounded Degree Representation (S-BDR) of order  $N$ , that is, for *almost every* pair  $(a, b) \in \mathbb{R}^n \times \mathbb{R}$

$$a^T x + b > 0 \text{ on } S \quad \Rightarrow \quad a^T x + b = \sum_{v \in \{0,1\}^m} \phi_v(x) g_v(x)$$

for some SOS polynomials  $\phi_v(x)$  with degree bounds  $\deg(\phi_v) + \deg(g_v) \leq 2N$ .

When  $S$  defined in (4.11) is compact (not necessarily convex), Lasserre [13] showed that the convex hull  $\text{conv}(S) = \tilde{S}_N$  if PP-BDR of order  $N$  holds for  $S$ ,  $\text{conv}(S) = \hat{S}_N$  if S-BDR of order  $N$  holds for  $S$ . This is summarized as follows.

**Theorem 4.11 (Theorem 2, Lasserre [13]).** *Suppose  $S$  is defined by (4.11) and is compact.*

- (a) *If PP-BDR of order  $N$  holds for  $S$ , then  $\text{conv}(S)$  equals  $\tilde{S}_N$  defined in (4.15).*
- (b) *If S-BDR of order  $N$  holds for  $S$ , then  $\text{conv}(S)$  equals  $\hat{S}_N$  defined in (4.16).*

*Proof.* Here we sketch the proof given by Lasserre in [13].

- (a) We have already seen in the construction of (4.15) that  $S \subset \tilde{S}_N$ , no matter if is convex or not. Since  $\tilde{S}_N$  is convex, it clearly holds that  $\text{conv}(S) \subset \tilde{S}_N$ . Now we prove the reverse containment by contradiction. Suppose there exist a  $\hat{y}$  satisfying the LMIs in (4.15) and  $\hat{x} = (\hat{y}_{e_1}, \dots, \hat{y}_{e_n}) \notin \text{conv}(S)$ . Since  $\text{conv}(S)$  is closed and convex, there exists a supporting hyperplane of  $\text{conv}(S)$  that excludes  $\hat{x}$ , i.e., there exists nonzero  $a \in \mathbb{R}^n$  and  $b \in \mathbb{R}$  such that

$$a^T x + b > 0, \quad \forall x \in S, \quad a^T \hat{x} + b < 0.$$

Since  $\text{conv}(S)$  is compact, we can choose the above  $(a, b)$  generically. By the property PP-BDR of order  $N$ , there exist SOS polynomials  $\sigma_0, \dots, \sigma_m$  such that

$$a^T x + b = \sigma_0(x) + \sigma_1(x)g_1(x) + \cdots + \sigma_m(x)g_m(x)$$

is an identity, with  $\deg(\sigma_i g_i) \leq 2N$ . For each  $i$ , we can find a symmetric  $W_i \geq 0$  making  $\sigma_i(x) = [x]_{N-d_i} W_i [x]_{N-d_i}^T$ . Here  $d_i = \lceil \deg(g_i)/2 \rceil$ . In the above identity, replace each monomial  $x^\alpha$  by  $\hat{y}_\alpha$ , then we get

$$a^T \hat{x} + b = \text{Trace}\left(L_{g_0}^{(N)}(\hat{y})W_0\right) + \cdots + \text{Trace}\left(L_{g_m}^{(N)}(\hat{y})W_m\right) \geq 0,$$

which contradicts the previous assertion  $a^T \hat{x} + b < 0$ . Therefore, we must have  $\text{conv}(S) = \tilde{S}_N$ .

- (b) Would be proved in almost the same way as (a). □

Critical in proving the main theorems of Sect. 4.3, Theorems 4.6 and 4.8, is the following which asserts that when every  $g_i(x)$  is either sos-concave or strictly concave on  $\partial S$ , the PP-BDR or P-BDR property holds for  $S$ . This is implied by the following theorem.

**Theorem 4.12.** Suppose  $S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}$  is compact convex and has nonempty interior. Assume each  $g_i(x)$  is concave on  $S$ .

- (i) Suppose the archimedean condition holds for  $S$ , and for each  $i$ , suppose either  $-\nabla^2 g_i(x)$  is SOS or  $-\nabla^2 g_i(u) > 0$  for all  $u \in \partial S_i \cap \partial S$ . Then there exists  $N > 0$  such that if a linear functional  $\ell^T x - \ell^*$  is nonnegative on  $S$ , we have

$$\ell^T x - \ell^* = \sum_{i=0}^m \sigma_i(x) g_i(x)$$

for some SOS polynomials  $\sigma_i$  satisfying  $\deg(\sigma_i g_i) \leq 2N$ . So, the PP-BDR of order  $N$  holds for  $S$ .

- (ii) Suppose for each  $i$ , either  $-\nabla^2 g_i(x)$  is SOS or  $-\nabla^2 g_i(u) > 0$  for all  $u \in \partial S_i \cap \partial S$ . Then there exists  $N > 0$  such that if a linear functional  $\ell^T x - \ell^*$  is nonnegative on  $S$ , we have

$$\ell^T x - \ell^* = \sum_{v \in \{0,1\}^m} \sigma_v(x) g_v(x)$$

for some SOS polynomials  $\sigma_v$  satisfying  $\deg(\sigma_v g_v) \leq 2N$ . So, the S-BDR of order  $N$  holds for  $S$ .

*Proof.* This theorem is a restatement of Theorem 20 in [7]. We refer to [7] for the proof.  $\square$

Combining Theorems 4.11 and 4.12, we can get the following conclusion.

**Theorem 4.13 (Theorems 5,6, [7]).** Suppose  $S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}$  is compact convex and has nonempty interior. Assume all  $g_i(x)$  are concave on  $S$ . Let  $\tilde{S}_N$  be defined in (4.15) and  $\hat{S}_N$  be defined in (4.16).

- (i) Suppose AC holds for  $S$ , and for each  $i$ , either  $-\nabla^2 g_i(x)$  is SOS or  $-\nabla^2 g_i(u) > 0$  for all  $u \in \partial S_i \cap \partial S$ . Then  $S = \tilde{S}_N$  for  $N$  big enough.  
(ii) Suppose for each  $i$ , either  $-\nabla^2 g_i(x)$  is SOS or  $-\nabla^2 g_i(u) > 0$  for all  $u \in \partial S_i \cap \partial S$ . Then  $S = \hat{S}_N$  for  $N$  big enough.

*Proof.* Clearly,  $S$  is a compact convex set, and then  $\text{conv}(S) = S$ . In (i), Theorem 4.12 implies PP-BDR holds for  $S$  of a certain order  $N$ . Then  $\tilde{S}_N = S$  follows Theorem 4.11. The proof of (ii) is the same.  $\square$

There are also some other conditions guaranteeing that (4.15) or (4.16) is a lifted LMI for  $S$ , like in Lasserre [14] and Nie [20]. Furthermore, for  $S$  to be representable by Putinar type moment construction (4.15), a necessary condition is that all faces of  $S$  are exposed. This is a pretty result of Netzer et al. [19].

It is possible that the defining polynomials  $g_i(x)$  are not concave but the set  $S$  is still convex. In this case, does  $S$  have an SDP representation? After some

modifications in LMI (4.16), the answer is affirmative in very general situations, as we saw in Theorem 4.6. If  $S$  is compact convex and its boundary has positive curvature, then using a chain of theorems we are now proving it must have an SDr. This will be shown in subsequent sections.

#### 4.6.2 Strictly Quasi-Concave Defining Polynomials

In this subsection, we prove some weaker sufficient conditions than sos-concavity or strict concavity to justify (4.15) or (4.16) is an SDP representation for  $S$ . We begin with a review of quasi-concavity.

A smooth function  $f$  on  $\mathbb{R}^n$  is said to be strictly quasi-concave at  $u$  if

$$-v^T \nabla^2 f(u)v > 0, \quad \forall 0 \neq v \in \nabla f(u)^\perp \quad (4.18)$$

where  $\nabla f(u)^\perp := \{v \in \mathbb{R}^n : \nabla f(u)^T v = 0\}$ . When  $\nabla f(u)$  vanishes, we require  $-\nabla^2 f(u) > 0$  in order for  $f(x)$  to be strictly quasi-concave at  $u$ . For a subset  $V \subset \mathbb{R}^n$ , we say  $f(x)$  is strictly quasi-concave on  $V$  if  $f(x)$  is strictly quasi-concave on every point on  $V$ . When  $>$  is replaced by  $\geq$  in (4.18), we can similarly define  $f(x)$  to be quasi-concave.

Suppose the set  $S$  is given in (4.11). It is possible that the individual set  $S_i = \{x \in \mathbb{R}^n : g_i(x) \geq 0\}$  is convex but that its defining polynomials  $g_i(x)$  are neither concave nor quasi-concave. Under some modest hypotheses, we have shown that  $S$  can be defined by a new set of polynomials having negative definite Hessians on  $S$  when every  $g_i$  is strictly quasi-concave.

The following proposition gives  $S_i$  a new defining polynomial  $p_i$  whose Hessian is negative definite on  $S$  when  $g_i(x)$  is strictly quasi-concave on  $S$ .

**Proposition 4.4 (Proposition 10, [7]).** *Assume  $g_i(x)$  is strictly quasi-concave on  $S$  defined in (4.11). Then there exists a polynomial  $h_i(x)$  such that  $h_i(x)$  is positive on  $S$  and the product  $p_i(x) = g_i(x)h_i(x)$  satisfies  $-\nabla^2 p_i(x) > 0$  on  $S$ .*

The main result of this section is

**Theorem 4.14.** *Suppose  $S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}$  is compact convex and has nonempty interior. If each  $g_i(x)$  is either sos-concave or strictly quasi-concave on  $S$ , then  $\hat{S}_N$  in (4.16) equals  $S$  for  $N$  sufficiently large. Furthermore, if in addition AC holds, then  $\tilde{S}_N$  in (4.15) equals  $S$  for  $N$  sufficiently large.*

Theorem 4.14 is stronger than Theorem 2 of [7], because it validates a concrete construction of SDr like (4.16) or (4.15). The proof of Theorem 2 in [7] is based on a different (also more complicated) construction which is (5.4) of [7]. The stronger conclusion and proof of Theorem 4.14 in the above is motivated by a personal communication of the authors with Markus Schweighofer, who pointed out this fact. Below we give the proof using his excellent ideas.

*Proof.* In our proof we shall give more details for the Schmüdgen type construction than for the Putinar construction, in order to expand the readers perspective beyond our previous exposition.

If  $-\nabla^2 g_i(x)$  is SOS in  $x$ , let  $p_i(x) = g_i(x)$  which is obviously concave. If  $g_i$  is strictly quasi-concave on  $S$ , let  $p_i(x)$  be the new defining polynomials for  $S_i$  given by Proposition 4.4, which have negative definite Hessian on  $S$ . For some small open set  $U$  containing  $S$ , the convex set  $S$  is equivalently defined as

$$S = \{x \in U : p_1(x) \geq 0, \dots, p_m(x) \geq 0\}.$$

As we have seen earlier,  $S$  is contained in the projection of LMI (4.16). We claim that this projection is sharp for  $N$  sufficiently large.

Otherwise, seeking a contradiction, suppose there exists a vector  $(\hat{x}, \hat{y})$  satisfying (4.16) such that  $\hat{x} \notin S$ . By the Hahn–Banach Separation Theorem, there must exist a vector  $\ell$  of unit length such that

$$\ell^T \hat{x} < \ell^* := \min_{x \in S} \ell^T x. \quad (4.19)$$

Let  $u \in S$  be the minimizer of  $\ell^T x$  on  $S$ , which must be on the boundary  $\partial S$ . Note that  $p_1(x), \dots, p_m(x)$  are concave polynomials, and  $S$  has nonempty interior. Since  $p_i(x) = h_i(x)g_i(x)$  for  $h_i(x)$  positive on  $S$ , the new equivalent constraints  $p_1(x) \geq 0, \dots, p_m(x) \geq 0$  also have nonempty interior. Thus the Slater's condition holds and hence there exist Lagrange multipliers  $\lambda_1, \dots, \lambda_m \geq 0$  such that

$$\ell = \sum_{i=1}^m \lambda_i \nabla p_i(u), \quad \lambda_i p_i(u) = 0, \quad \forall i = 1, \dots, m.$$

Now we focus on LMI (4.16). By Theorem 4.12, we get

$$\tilde{\ell}_t(x) = \ell^T x - \ell^* - \sum_{i=1}^m \lambda_i p_i(x) = \sum_{v \in \{0,1\}^m} \sigma_v(x) g_1^{v_1}(x) \cdots g_m^{v_m}(x)$$

for some SOS polynomials  $\sigma_v$  with  $\deg(\sigma_v g_1^{v_1} \cdots g_m^{v_m}) \leq 2N$ , for some integer  $N$ . So we have the identity

$$\ell^T x - \ell^* = \sum_{i=1}^m \lambda_i p_i(x) + \sum_{v \in \{0,1\}^m} \sigma_v(x) g_1^{v_1}(x) \cdots g_m^{v_m}(x).$$

Note that every  $p_i(x) = h_i(x)g_i(x)$  and  $h_i(x)$  is strictly positive on  $S$ . By Schmüdgen's Positivstellensatz, there exists SOS polynomials  $s_v^{(i)}$  such that

$$h_i(x) = \sum_{v \in \{0,1\}^m} s_v^{(i)}(x) g_1^{v_1}(x) \cdots g_m^{v_m}(x).$$

Thus, without loss of generality, we have the identity

$$\ell^T x - \ell^* = \sum_{\nu \in \{0,1\}^m} \sigma_\nu(x) g_1^{\nu_1}(x) \cdots g_m^{\nu_m}(x).$$

with every  $\sigma_\nu(x)$  being SOS and having degree bounded by  $N$ . Note that  $N$  is still independent of  $\ell$  and  $(\hat{x}, \hat{y})$ . We can write  $\sigma_\nu(x) = [x]_{d-d_\nu}^T W_\nu [x]_{d-d_\nu}$  for some symmetric matrix  $W_\nu \geq 0$ . In the identity above, replacing each monomial  $x^\alpha$  by  $\hat{y}_\alpha$ , we get

$$\ell^T \hat{x} - \ell^* = \sum_{\nu \in \{0,1\}^m} \text{Trace}\left(W_\nu \cdot \left(\sum_{0 \leq |\alpha| \leq 2N} A_\alpha^\nu \hat{y}_\alpha\right)\right) \geq 0,$$

which contradicts (4.19). This completes the proof that (4.16) is an SDr for  $S$ .

Next we turn to LMI (4.15). If, in addition, the archimedean condition holds, we similarly prove (4.15) is also a lifted LMI for  $S$  for big  $N$ . The proof is almost the same as above. The only difference is to obtain an identity like

$$\ell^T x - \ell^* = \sum_{i=0}^m \sigma_i(x) g_i(x)$$

with every  $\sigma_i$  being SOS and  $\deg(\sigma_i g_i) \leq 2N$  ( $N$  is also independent of  $\ell$ ).  $\square$

## 4.7 Convex Hull and Localization

A different but very useful technique for constructing an SDr is localization, that is, finding an SDr for local parts of a set. After getting all local ones, we group them together into a single SDr via constructing the convex hull of unions. This technique has already been used in Sect. 4.3.2, but now we give more detail. We begin with an SDr for convex hulls.

### 4.7.1 SDr for Convex Hulls

Let  $W_1, \dots, W_m \subset \mathbb{R}^n$  be convex sets. Then their Minkowski sum

$$W_1 + \cdots + W_m = \{x = x_1 + \cdots + x_m : x_1 \in W_1, \dots, x_m \in W_m\}$$

is also a convex set. If every  $W_k$  has an SDr, then an SDr for  $W_1 + \cdots + W_m$  can also be obtained by definition. Usually the union of convex sets  $W_1, \dots, W_m$  is no longer convex, but its convex hull  $\text{conv}(\cup_{k=1}^m W_k)$  is. We give a representation for  $\text{conv}(\cup_{k=1}^m W_k)$  first.

**Lemma 4.3 (Lemma 2.1, [8]).** *If  $W_k$  are all nonempty convex sets, then*

$$\text{conv}\left(\bigcup_{k=1}^m W_k\right) = \bigcup_{\lambda \in \Delta_m} (\lambda_1 W_1 + \cdots + \lambda_m W_m)$$

where  $\Delta_m = \{\lambda \in \mathbb{R}_+^m : \lambda_1 + \cdots + \lambda_m = 1\}$  is the standard simplex.

Based on Lemma 4.3, it is possible to construct an SDr for the convex hull  $\text{conv}(\bigcup_{k=1}^m W_k)$  from the ones for  $W_k$ . This is summarized in the following theorem, already foreshadowed by Theorem 4.7 in Sect. 4.3.

**Theorem 4.15 (Theorem 2.2, [8]).** *Let  $W_1, \dots, W_m$  be nonempty convex sets given by SDP representations*

$$W_k = \left\{ x \in \mathbb{R}^n : \exists u^{(k)}, A^{(k)} + \sum_{i=1}^n x_i B_i^{(k)} + \sum_{j=1}^{N_k} u_j^{(k)} C_j^{(k)} \succeq 0 \right\}$$

for some symmetric matrices  $A^{(k)}, B_i^{(k)}, C_j^{(k)}$ . Define a new set

$$C = \left\{ \sum_{k=1}^m x^{(k)} : \exists \lambda \in \Delta_m, \exists u^{(k)}, \lambda_k A^{(k)} + \sum_{i=1}^n x_i^{(k)} B_i^{(k)} + \sum_{j=1}^{N_k} u_j^{(k)} C_j^{(k)} \succeq 0, 1 \leq k \leq m \right\}. \quad (4.20)$$

Then we have

$$\text{conv}\left(\bigcup_{k=1}^m W_k\right) \subseteq C, \quad \overline{C} = \overline{\text{conv}\left(\bigcup_{k=1}^m W_k\right)}. \quad (4.21)$$

In addition, if every  $W_k$  is bounded, then

$$C = \text{conv}\left(\bigcup_{k=1}^m W_k\right). \quad (4.22)$$

When some  $W_k$  is unbounded,  $C$  and  $\text{conv}(\bigcup_{k=1}^m W_k)$  might not be equal, but they have the same interior, which is often good enough in optimization. When some  $W_k$  is unbounded,  $C$  and  $\text{conv}(\bigcup_{k=1}^m W_k)$  might not be equal, and  $C$  might not be closed. The following examples illustrate this.

*Example 4.8.* (i) Consider  $W_1 = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} x_1 & 1 \\ 1 & x_2 \end{bmatrix} \succeq 0 \right\}$ ,  $W_2 = \{0\}$ . The convex hull

$$\text{conv}(W_1 \cup W_2) = \{x \in \mathbb{R}_+^2 : x_1 + x_2 = 0 \text{ or } x_1 x_2 > 0\}.$$

However, it holds that

$$C = \left\{ x \in \mathbb{R}^2 : \exists \lambda \geq 0, \begin{bmatrix} x_1 & \lambda_1 \\ \lambda_1 & x_2 \end{bmatrix} \succeq 0 \right\} = \mathbb{R}_+^2.$$

So,  $C$  and  $\text{conv}(W_1 \cup W_2)$  are not equal.

(ii) Consider  $W_1 = \left\{ x \in \mathbb{R}^2 : \exists u \geq 0, \begin{bmatrix} x_1 & 1+x_2 \\ 1+x_2 & 1+u \end{bmatrix} \succeq 0 \right\}$  and  $W_2 = \{0\}$ . It holds that

$$\begin{aligned} \text{conv}(W_1 \cup W_2) &= \{x \in \mathbb{R}^2 : x_1 > 0, \text{ or } x_1 = 0 \text{ and } -1 \leq x_2 \leq 0\}, \\ \overline{\text{conv}(W_1 \cup W_2)} &= \{x \in \mathbb{R}^2 : x_1 \geq 0\}. \end{aligned}$$

We can verify that  $C = \text{conv}(W_1 \cup W_2)$ , but it is not closed.

Now we show some examples that the boundedness of  $W_1, \dots, W_m$  is not necessary for (4.22) to hold.

*Example 4.9.* (a) Consider the special case where each  $W_k$  is homogeneous, i.e.,  $A^{(k)} = 0$  in the SDP representation of  $W_k$ . Then by Lemma 4.3, we immediately have

$$C = \text{conv}\left(\bigcup_{k=1}^m W_k\right).$$

(b) Consider

$$W_1 = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -x_1 & 1 \\ 1 & x_2 \end{bmatrix} \succeq 0 \right\}, W_2 = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} x_1 & 1 \\ 1 & x_2 \end{bmatrix} \succeq 0 \right\}$$

It can be verified that  $\text{conv}(W_1 \cup W_2)$  is given by

$$\begin{aligned} C &= \left\{ x + y \in \mathbb{R}^2 : \exists \lambda \in \mathcal{A}_2, \begin{bmatrix} -x_1 & \lambda_1 \\ \lambda_1 & x_2 \end{bmatrix} \succeq 0, \begin{bmatrix} y_1 & \lambda_2 \\ \lambda_2 & y_2 \end{bmatrix} \succeq 0 \right\} \\ &= \{x \in \mathbb{R}^2 : x_2 > 0\}. \end{aligned} \tag{4.23}$$

#### 4.7.2 SDr Via Localization

We now use the SDr construction for convex hulls in Theorem 4.15 to indicate the proof of Theorem 4.6, for SDP representability of a convex basic semialgebraic set  $S$ . We can use conditions on the defining polynomials on the part of

the boundary of  $S$  instead of requiring they hold on the whole set  $S$ . The idea of localization is captured by the following proposition.

**Proposition 4.5.** *Let  $S$  be a compact convex set. Then  $S$  is SDr if and only if for every  $u \in \partial S$ , there exists some  $\delta > 0$  such that  $S \cap \bar{B}(u, \delta)$  is SDr.*

*Proof.* “ $\Rightarrow$ ” Suppose  $S$  has SDP representation

$$S = \left\{ x \in \mathbb{R}^n : A + \sum_{i=1}^n x_i B_i + \sum_{j=1}^N u_j C_j \succeq 0 \right\}$$

for symmetric matrices  $A, B_i, C_j$ . Then  $S \cap \bar{B}(u, \delta)$  also has SDP representation

$$\left\{ x \in \mathbb{R}^n : A + \sum_{i=1}^n x_i B_i + \sum_{j=1}^N u_j C_j \succeq 0, \quad \begin{bmatrix} I_n & x-u \\ (x-u)^T & \delta^2 \end{bmatrix} \succeq 0 \right\}.$$

“ $\Leftarrow$ ” Suppose for every  $u \in \partial S$  the set  $S \cap \bar{B}(u, \delta_u)$  has SDP representation for some  $\delta_u > 0$ . Note that  $\{B(u, \delta_u) : u \in \partial S\}$  is an open cover for the compact set  $\partial S$ . So there are a finite number of balls, say,  $B(u_1, \delta_1), \dots, B(u_L, \delta_L)$ , to cover  $\partial S$ . Note that

$$\begin{aligned} S &= \text{conv}(\partial S) = \text{conv}\left(\bigcup_{k=1}^L (\partial S \cap \bar{B}(u_k, \delta_k))\right) \\ &\subseteq \text{conv}\left(\bigcup_{k=1}^L (S \cap \bar{B}(u_k, \delta_k))\right) \subseteq S. \end{aligned} \tag{4.24}$$

The sets  $S \cap \bar{B}(u_k, \delta_k)$  are all bounded. By Theorem 4.15, we know

$$S = \text{conv}\left(\bigcup_{k=1}^L S \cap \bar{B}(u_k, \delta_k)\right)$$

has SDP representation.  $\square$

When the set  $S$  is basic closed semialgebraic, we prove the following sufficient condition for SDP representability, which is close to Theorem 4.6, next we prove Theorem 4.6.

**Theorem 4.16.** *Assume  $S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}$  is a compact convex set defined by polynomials  $g_i$  and has nonempty interior. If for every  $u \in \partial S$  and  $i$  for which  $g_i(u) = 0$ ,  $g_i$  is either sos-concave or strictly quasi-concave at  $u$ , then  $S$  is SDP representable.*

*Proof.* For any  $u \in \partial S$ , let  $I(u) = \{1 \leq i \leq m : g_i(u) = 0\}$  denote the active set of constraints at  $u$ . For every  $i \in I(u)$ , if  $g_i(x)$  is not sos-concave,  $g_i(x)$  is strictly

quasi-concave at  $u$ . By continuity, there exist some  $\delta > 0$  such that  $g_i(x)$  is strictly quasi-concave on  $\bar{B}(u, \delta)$ . Note  $g_i(u) > 0$  for  $i \notin I(u)$ . So we can choose  $\delta > 0$  small enough such that

$$g_i(x) > 0, \forall i \notin I(u), \quad \forall x \in \bar{B}(u, \delta).$$

Therefore, the set  $S_u := S \cap \bar{B}(u, \delta)$  can be defined equivalently by only using active  $g_i$ , namely,

$$S_u = \left\{ x \in \mathbb{R}^n : g_i(x) \geq 0, \forall i \in I(u), \delta^2 - \|x - u\|^2 \geq 0 \right\}.$$

For every  $i \in I(u)$ , the defining polynomial  $g_i(x)$  is either sos-concave or strictly quasi-concave on  $S_u$ . Obviously  $S_u$  is a compact convex set with nonempty interior. By Theorem 4.14,  $S_u$  is SDP representable. We can do this for a finite cover of  $\partial S$ , and hence by Proposition 4.5,  $S$  is also SDP representable.  $\square$

*Proof.* The condition  $g_i$  is strictly quasi-concave at  $u$  clearly implies the hypothesis  $Z_i$  has positive curvature at any  $u$  where it is nonsingular. However, there is a difference between the hypotheses of Theorems 4.6 and 4.16 in that Theorem 4.6 specifies varieties  $Z_i$  while Theorem 4.16 specifies a particular defining polynomial for each  $Z_i$ . The trouble is that  $\nabla g_i(u)$  might equal 0 at some relevant  $u \in Z_i$ . However, one can prove Theorem 4.6, because if  $\nabla g_i(u) = 0$  and the variety  $Z_i$  is nonsingular another defining polynomial  $\tilde{g}_i$  for  $Z_i$  will be nonsingular, hence quasi-concave at  $u$ . Theorem 4.16 applies to  $\tilde{g}_i$ , and this coupled with localization yields Theorem 4.6. Note that even if we must use several defining polynomials  $\tilde{g}_i$  for  $Z_i$ , we can pick balls to localize each one appropriately.  $\square$

**Acknowledgements** J. William Helton was partially supported by NSF grants DMS-0757212, DMS-0700758 and the Ford Motor Company. Jiawang Nie was partially supported by NSF grants DMS-0757212, DMS-0844775 and Hellman Foundation Fellowship.

## Appendix

### Positive Curvature

This section gives more details and perspectives on the notion of positive curvature.

Let  $S = \{x \in \mathbb{R}^n : g_1(x) \geq 0, \dots, g_m(x) \geq 0\}$  be a basic closed semialgebraic set; here the  $g_i(x)$  are in the ring  $\mathbb{R}[x]$  of multivariate polynomials with real coefficients and are called the defining polynomials for  $S$ . Assume  $S$  is convex, compact and has nonempty interior. Let  $S_i = \{x \in \mathbb{R}^n : g_i(x) \geq 0\}$  and  $Z(g_i) = \{x \in \mathbb{R}^n : g_i(x) = 0\}$  be the zero set of  $g_i$ . Denote by  $\partial S$  and  $\partial S_i$  the boundaries of  $S$  and  $S_i$  respectively. Note that  $\partial S_i$  might be contained in  $Z(g_i)$  properly.

We first review the definition of curvature. For a smooth function  $f(x)$  on  $\mathbb{R}^n$ , suppose the zero set  $Z(f) := \{x \in \mathbb{R}^n : f(x) = 0\}$  is nonsingular at a point  $u \in Z(f)$ , i.e.,  $\nabla f(u) \neq 0$ . Then  $Z(f)$  is a smooth hypersurface near the point  $u$ .  $Z(f)$  is said to

have positive curvature at the nonsingular point  $u \in Z(f)$  if its second fundamental form is positive definite, i.e.,

$$-v^T \nabla^2 f(u)v > 0, \quad \forall 0 \neq v \in \nabla f(u)^\perp \quad (4.25)$$

where  $\nabla f(u)^\perp := \{v \in \mathbb{R}^n : \nabla f(u)^T v = 0\}$ . For a subset  $V \subset Z(f)$ , we say  $Z(f)$  has positive curvature on  $V$  if  $f(x)$  is nonsingular on  $V$  and  $Z(f)$  has positive curvature at every  $u \in V$ . When  $>$  is replaced by  $\geq$  in (4.25), we can similarly define  $Z(f)$  has nonnegative curvature at  $u$ . We emphasize that this definition applies to any zero sets defined by smooth functions on their nonsingular points. We refer to Spivak [29] for more on curvature and the second fundamental form.

The sign “ $-$ ” in the front of (4.25) might look confusing for some readers, since  $Z(f)$  and  $Z(-f)$  define exactly the same zero set. Geometrically, the curvature of a hypersurface should be independent of the sign of the defining functions. The reason for including the minus sign in (4.25) is we are interested in the case where the set  $\{x : f(x) \geq 0\}$  is locally convex near  $u$  when  $Z(f)$  has positive curvature at  $u$ . Now we give more geometric perspective by describing alternative formulations of positive curvature. Geometrically, the zero set  $Z(f)$  has nonnegative (resp. positive) curvature at a nonsingular point  $u \in Z(f)$  if and only if there exists an open set  $O_u$  such that  $Z(f) \cap O_u$  can be represented as the graph of a function  $\phi$  which is (strictly) convex at the origin in an *appropriate* coordinate system (see Ghomi [6]). Here we define a function to be convex (resp. strictly convex) at some point if its Hessian is positive semidefinite (resp. definite) at that point.<sup>3</sup> Also note when  $Z(f)$  has positive curvature at  $u$ , the set  $\{x : f(x) \geq 0\}$  is locally convex near  $u$  if and only if (4.25) holds, or equivalently the set  $\{x : f(x) \geq 0\} \cap O_u$  is above the graph of function  $\phi$ . Now we prove the statements above and show such  $\phi$  exists. When the gradient  $\nabla f(u) \neq 0$ , by the Implicit Function Theorem, in an open set near  $u$  the hypersurface  $Z(f)$  can be represented as the graph of some smooth function in a certain coordinate system. Suppose the origin of this coordinate system corresponds to the point  $u$ , and the set  $\{x : f(x) \geq 0\}$  is locally convex near  $u$ . Let us make the affine linear coordinate transformation

$$x - u = [\nabla f(u) \ G(u)]^T \begin{bmatrix} y \\ x' \end{bmatrix} \quad (4.26)$$

where  $(y, x') \in \mathbb{R} \times \mathbb{R}^{n-1}$  are new coordinates and  $G(u)$  is an orthogonal basis for subspace  $\nabla f(u)^\perp$ . By the Implicit Function Theorem, since  $\nabla f(u) \neq 0$ , in some neighborhood  $O_u$  of  $u$ , the equation  $f(x) = 0$  defines a smooth function  $y = \phi(x')$ . For simplicity, we reuse the letter  $f$  and write  $f(x', y) = f(x', \phi(x')) = 0$ . Since  $\nabla f(u)$  is orthogonal to  $G(u)$ , we have  $f_y(0, 0) = \|\nabla f(u)\|^2$  and  $\nabla_{x'} \phi(0) = 0$ . Twice differentiating  $f(x', y) = 0$  gives

$$\nabla_{x'x'} f + \nabla_{x'y} f \nabla_{x'} \phi^T + \nabla_{x'} \phi \nabla_{x'y} f^T + f_{yy} \nabla_{x'} \phi \nabla_{x'} \phi^T + f_y \nabla_{x'x'} \phi = 0.$$

---

<sup>3</sup>In general,  $f$  being strictly convex does not imply its Hessian is positive definite.

Evaluate the above at the origin in the new coordinates  $(y, x')$ , to get

$$\nabla_{x'x'}\phi(0) = -\frac{1}{\|\nabla f(u)\|^2} \nabla_{x'x'} f(u).$$

So we can see  $Z(f)$  has positive (resp. nonnegative) curvature at  $u$  if and only if the function  $y = \phi(x')$  is strictly convex (resp. convex) at  $u$ . Since at  $u$  the direction  $\nabla f(u)$  points to the inside of the set  $\{x : f(x) \geq 0\}$ , the intersection  $\{x : f(x) \geq 0\} \cap O_u$  lies above the graph of  $\phi$ .

The notion of positive curvature of a nonsingular hypersurface  $Z(f)$  does not distinguish one side of  $Z(f)$  from the other. For example, the boundary of the unit ball  $\bar{B}(0, 1)$  is the unit sphere, a manifold with positive curvature by standard convention. However,  $\bar{B}(0, 1)$  can be expressed as  $\{x : f(x) \geq 0\}$  where  $f(x) = 1 - \|x\|^2$ , or equivalently as  $\{x : h(x) \leq 0\}$  where  $h(x) = \|x\|^2 - 1$ . Note that  $Z(f) = Z(h)$ , but  $-\nabla^2 f(x) > 0$  and  $+\nabla^2 h(x) > 0$ .

However, on a nonsingular hypersurface  $Z(f)$  one can designate its sides by choosing one of the two normal directions  $\pm\nu(x)$  at points  $x$  on  $Z(f)$ . We call one such determination at some point, say  $u$ , the *outward direction*, and then select, at each  $x$ , the continuous function  $\nu(x)$  to be consistent with this determination. In the ball example,  $\nabla f(x) = -2x$  and we would typically choose  $\nu(x) = -\nabla f(x)$  to be the outward normal direction to  $Z(f)$ . In the more general case described below (4.26), let us call  $-\nabla f(x)$  the outward normal, which near the origin points away from the set  $\{(x', y) : y \geq \phi(x')\}$ . To see this, note that  $-\nabla f(0, 0) = [0 \ -\|\nabla f(u)\|^2]^T$ .

We remark that the definition of positive curvature for some hypersurface  $Z$  at a nonsingular point is independent of the choice of defining functions. Suppose  $f$  and  $g$  are smooth defining functions such that  $Z \cap B(u, \delta) = Z(f) \cap B(u, \delta) = Z(g) \cap B(u, \delta)$ ,  $\nabla f(u) \neq 0 \neq \nabla g(u)$  for some  $\delta > 0$  and

$$\{x \in B(u, \delta) : f(x) \geq 0\} = \{x \in B(u, \delta) : g(x) \geq 0\}. \quad (4.27)$$

Then the second fundamental form in terms of  $f$  is positive definite (resp. semidefinite) at  $u$  if and only if the second fundamental form in terms of  $g$  is positive definite (resp. semidefinite) at  $u$ . To see this, note that  $\nabla f(u) = \alpha \nabla g(u)$  for some scalar  $\alpha \neq 0$ , because  $\nabla f(u)$  and  $\nabla g(u)$  are perpendicular to the boundary of  $Z$  at  $u$ . Also  $\alpha > 0$  because of (4.27). Then in the new coordinate system  $(y, x')$  defined in (4.26), as we have seen earlier,  $Z$  has nonnegative (resp. positive) curvature at  $u$  if and only if the function  $y = \phi(x')$  is convex (resp. strictly convex) at  $u$ , which holds if and only if either one of  $f$  and  $g$  has positive definite (resp. semidefinite) second fundamental form. So the second fundamental form of  $f$  and  $g$  are simultaneously positive definite or semidefinite.

## References

1. Ahmadi, A.A., Parrilo, P.A.: A convex polynomial that is not sos-convex. To appear in Mathematical Programming
2. Blekherman, G.: Convex forms that are not sums of squares. Preprint (2009). <http://arxiv.org/abs/0910.0656>
3. Bochnak, J., Coste, M., Roy, M.-F.: Real Algebraic Geometry, Springer (1998)
4. Branden, P.: Obstructions to determinantal representability. Advances in Mathematics **226**, 1202–1212 (2011)
5. Conway, J.B.: A course in Functional Analysis, Springer Graduate Texts in Math Series (1985)
6. Ghomi, M.: Optimal Smoothing for Convex Polytopes. Bull. London Math. Soc. **36**, 483–492 (2004)
7. Helton, J.W., Nie, J.: Semidefinite representation of convex sets. Mathematical Programming **122**(1), 21–64 (2010)
8. Helton, J.W., Nie, J.: Sufficient and Necessary Conditions for Semidefinite Representability of Convex Hulls and Sets. SIAM Journal on Optimization **20**(2), 759–791 (2009)
9. Helton, J.W., Nie, J.: Structured Semidefinite Representation of Some Convex Sets. Proceedings of 47th IEEE Conference on Decision and Control (CDC), pp. 4797–4800, Cancun, Mexico, 9-11 December 2008
10. Helton, J.W., McCullough, S., Vinnikov, V.: Noncommutative convexity arises from linear matrix inequalities. J. Funct. Anal. **240**(1), 105–191 (2006)
11. Helton, W., Vinnikov, V.: Linear matrix inequality representation of sets. Comm. Pure Appl. Math **60**(5), 654–674 (2007)
12. Henrion, D.: Semidefinite representation of convex hulls of rational varieties. LAAS-CNRS Research Report No. 09001, January 2009
13. Lasserre, J.: Convex sets with semidefinite representation. Mathematical Programming **120**, 457–477 (2009)
14. Lasserre, J.: Convexity in SemiAlgebraic Geometry and Polynomial Optimization. SIAM J. Optim. **19**, 1995–2014 (2009)
15. Lax, P.: Differential equations, difference equations and matrix theory. Communications on Pure and Applied Mathematics **6**, 175–194 (1958)
16. Lewis, A., Parrilo P., Ramana, M.: The Lax conjecture is true. Proc. Amer. Math. Soc. **133**(9), 2495–2499 (2005)
17. Nesterov, Y., Nemirovskii, A.: Interior-point polynomial algorithms in convex programming. SIAM Studies in Applied Mathematics 13. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA (1994)
18. Nemirovski, A.: Advances in convex optimization: conic programming. Plenary Lecture at the International Congress of Mathematicians (ICM), Madrid, Spain, 22-30 August 2006
19. Netzer, T., Plaumann, D., Schweighofer, M.: Exposed faces of semidefinitely representable sets. SIAM Journal on Optimization **20**(4), 1944–1955 (2010)
20. Nie, J.: First Order Conditions for Semidefinite Representations of Convex Sets Defined by Rational or Singular Polynomials. To appear in Mathematical Programming
21. Parrilo, P.: Exact semidefinite representation for genus zero curves. Paper presented at the Positive Polynomials and Optimization Workshop, Banff, Canada, 7–12 October 2006
22. Parrilo, P., Sturmfels, B.: Minimizing polynomial functions. In: Basu, S., Gonzalez-Vega, L. (eds.) Proceedings of the DIMACS Workshop on Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science, March 2001. pp. 83–100. American Mathematical Society (2003)
23. Putinar, M.: Positive polynomials on compact semi-algebraic sets, Ind. Univ. Math. J. **42** 203–206 (1993)

24. Ranestad, K., Sturmfels, B.: On the convex hull of a space curve. *Advances in Geometry*, To appear in *Advances in Geometry*
25. Ranestad, K., Sturmfels, B.: The convex hull of a variety. In: *Notions of Positivity and the Geometry of Polynomials* Branden, P., Passare, M., Putinar, M. (eds.), Trends in Mathematics, Springer Verlag, Basel, pp. 331–344, (2011)
26. Reznick, B.: Some concrete aspects of Hilbert’s 17<sup>th</sup> problem. In: *Contemp. Math.*, vol. 253, pp. 251–272. American Mathematical Society (2000)
27. Sanyal, R., Ottobre, F., Sturmfels, B.: Orbitopes. *Mathematika* **57**, 275–314 (2011)
28. Scheiderer, C.: Convex hulls of curves of genus one. arXiv:1003.4605v1 [math.AG]
29. Spivak, M.: A comprehensive introduction to differential geometry, vol. II, second edition, Publish or Perish, Inc., Wilmington, Del. (1979)

# Chapter 5

## Convex Hulls of Algebraic Sets

João Gouveia and Rekha Thomas

### 5.1 Introduction

An important concern in optimization is the complete or partial knowledge of the convex hull of the set of feasible solutions to an optimization problem. Computing convex hulls is in general a difficult task, and a classical example is the construction of the integer hull of a polyhedron which drives many algorithms in integer programming. In this article we describe a method to convexify (at least approximately), an algebraic set using semidefinite programming.

By an algebraic set we mean a subset  $S \subseteq \mathbb{R}^n$  described by a finite list of polynomial equations of the form  $f_1(\mathbf{x}) = f_2(\mathbf{x}) = \dots = f_t(\mathbf{x}) = 0$  where  $f_i(\mathbf{x})$  is an element of  $\mathbb{R}[\mathbf{x}] := \mathbb{R}[x_1, \dots, x_n]$ , the polynomial ring in  $n$  variables over the reals. The input to our algorithm is the ideal generated by  $f_1, \dots, f_t$ , denoted as  $I = \langle f_1, \dots, f_t \rangle$ , which is the set  $\{\sum_{i=1}^t g_i f_i : g_i \in \mathbb{R}[\mathbf{x}]\}$ . An ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$  is a group under addition and is closed under multiplication by elements of  $\mathbb{R}[\mathbf{x}]$ . Given an ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$ , its *real variety*,  $\mathcal{V}_{\mathbb{R}}(I) = \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) = 0 \forall f \in I\}$  is an example of an algebraic set. Given  $I$ , we describe a method to produce a nested sequence of convex relaxations of the closure of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ , the convex hull of  $\mathcal{V}_{\mathbb{R}}(I)$ , called

---

J. Gouveia (✉)

Department of Mathematics, University of Washington, Box 354350, Seattle,  
WA 98195-4350, USA

CMUC, Department of Mathematics, University of Coimbra, 3001-454 Coimbra, Portugal  
e-mail: [igouveia@mat.uc.pt](mailto:igouveia@mat.uc.pt)

R. Thomas

Department of Mathematics, University of Washington, Box 354350, Seattle,  
WA 98195-4350, USA  
e-mail: [thomas@math.washington.edu](mailto:thomas@math.washington.edu)

the *theta bodies* of  $I$ . The  $k$ -th theta body  $\text{TH}_k(I)$  is obtained as the projection of a *spectrahedron* (the feasible region of a semidefinite program), and

$$\text{TH}_1(I) \supseteq \text{TH}_2(I) \supseteq \cdots \supseteq \text{TH}_k(I) \supseteq \text{TH}_{k+1}(I) \supseteq \cdots \supseteq \text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I))).$$

Of special interest to us are *real radical ideals*. We define the real radical of an ideal  $I$ , denoted as  $\sqrt{\mathbb{R}I}$ , to be the set of all polynomials  $f \in \mathbb{R}[\mathbf{x}]$  such that  $f^{2m} + \sum g_i^2 \in I$  for some  $g_i \in \mathbb{R}[\mathbf{x}]$  and  $m \in \mathbb{N}$ . We say that  $I$  is a real radical ideal if  $I = \sqrt{\mathbb{R}I}$ . Given a set  $S \subseteq \mathbb{R}^n$ , the *vanishing ideal* of  $S$ , denoted as  $\mathcal{I}(S)$ , is the set of all polynomials in  $\mathbb{R}[\mathbf{x}]$  that vanish on  $S$ . The *Real Nullstellensatz* (see Theorem 5.2) says that for any ideal  $I$ ,  $\mathcal{I}(\mathcal{V}_{\mathbb{R}}(I)) = \sqrt{\mathbb{R}I}$ .

The construction of theta bodies for arbitrary ideals was motivated by a problem posed by Lovász. In [17] Lovász constructed the theta body of a graph, a convex relaxation of the stable set polytope of a graph which was shown later to have a description in terms of semidefinite programming. An important result in this context is that the theta body of a graph coincides with the stable set polytope of the graph if and only if the graph is perfect. Lovász observed that the theta body of a graph could be described in terms of sums of squares of real polynomials modulo the ideal of polynomials that vanish on the incidence vectors of stable sets. This observation naturally suggests the definition of a theta body for any ideal in  $\mathbb{R}[\mathbf{x}]$ . In fact, an easy extension of his observation leads to a hierarchy of theta bodies for all ideals as above. In [18, Problem 8.3], Lovász asked to characterize all ideals that have the property that their first theta body coincides with  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$ , which was the starting point of our work. For defining ideals of finite point sets we answer this question in Sect. 5.4.

This article is organized as follows. In Sect. 5.2 we define theta bodies of an ideal in  $\mathbb{R}[\mathbf{x}]$  in terms of sums of squares polynomials. For a general ideal  $I$ , we get that  $\text{TH}_k(I)$  contains the closure of the projection of a spectrahedron which is described via combinatorial moment matrices from  $I$ . When the ideal  $I$  is real radical, we show that  $\text{TH}_k(I)$  coincides with the closure of the projected spectrahedron, and when  $I$  is the defining ideal of a set of points in  $\{0, 1\}^n$ , the closure is not needed. We establish a general relationship between the theta body sequence of an ideal  $I$  and that of its real radical ideal  $\sqrt{\mathbb{R}I}$ .

Section 5.3 gives two examples of the construction described in Sect. 5.2. As our first example, we look at the stable sets in a graph and describe the hierarchy of theta bodies that result. The first member of the hierarchy is Lovász's theta body of a graph. This hierarchy converges to the stable set polytope in finitely many steps as is always the case when we start with a finite set of real points. The second example is a cardioid in the plane in which case the algebraic set that is being convexified is infinite.

In Sect. 5.4 we discuss convergence issues for the theta body sequence. When  $\mathcal{V}_{\mathbb{R}}(I)$  is compact, the theta body sequence is guaranteed to converge to the closure of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  asymptotically. We prove that when  $\mathcal{V}_{\mathbb{R}}(I)$  is finite,  $\text{TH}_k(I) = \text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$  for some finite  $k$ . In the case of finite convergence, it is useful to know the specific value of  $k$  for which  $\text{TH}_k(I) = \text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$ . This notion is called exactness and we characterize exactness for the first theta body when the set to be convexified is finite. There are examples in which the theta body sequence does

not converge to  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$ . While a full understanding of when convergence occurs is still elusive, we describe one obstruction to finite convergence in terms of certain types of singularities of  $\mathcal{V}_{\mathbb{R}}(I)$ .

The last section gives more examples of theta bodies and their exactness. In particular we consider cuts in a graph and polytopes coming from the graph isomorphism question.

The core of this chapter is based on results from [5] and [3] which are presented here with a greater emphasis on geometry, avoiding some of the algebraic language in the original results. Theorems 5.5, 5.7 and their corollaries are new while Theorem 5.9 is from [4]. The application of theta bodies to polytopes that arise in the graph isomorphism question is taken from [15].

## 5.2 Theta Bodies of Polynomial Ideals

To describe the convex hull of an algebraic set, we start with a simple observation about any convex hull. Given a set  $S \subseteq \mathbb{R}^n$ ,  $\text{cl}(\text{conv}(S))$ , the closure of  $\text{conv}(S)$ , is the intersection of all closed half-spaces containing  $S$ :

$$\text{cl}(\text{conv}(S)) = \{\mathbf{p} \in \mathbb{R}^n : l(\mathbf{p}) \geq 0 \ \forall l \in \mathbb{R}[\mathbf{x}]_1 \text{ s.t. } l|_S \geq 0\}.$$

From a computational point of view, this observation is useless, as the right hand side is hopelessly cumbersome. However, if  $S$  is the zero set of an ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$ , we can define nice relaxations of the above intersection of infinitely many half-spaces using a classical strengthening of the nonnegativity condition  $l|_S \geq 0$ . We describe these relaxations in this section. In Sect. 2.1 we introduce our method for arbitrary ideals in  $\mathbb{R}[\mathbf{x}]$ . In Sect. 2.2 we specialize to real radical ideals, which occur frequently in applications, and show that in this case, much stronger results hold than for general ideals.

### 5.2.1 General Ideals

Recall that given an ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$ , two polynomials  $f$  and  $g$  are defined to be congruent modulo  $I$ , written as  $f \equiv g \pmod{I}$ , if  $f - g \in I$ . The relation  $\equiv$  is an equivalence relation on  $\mathbb{R}[\mathbf{x}]$  and the equivalence class of a polynomial  $f$  is denoted as  $f + I$ . The set of all congruence classes of polynomials modulo  $I$  is denoted as  $\mathbb{R}[\mathbf{x}]/I$  and this set is both a ring and a  $\mathbb{R}$ -vector space: given  $f, g \in \mathbb{R}[\mathbf{x}]$  and  $\lambda \in \mathbb{R}$ ,  $(f + I) + (g + I) = (f + g) + I$ ,  $\lambda(f + I) = \lambda f + I$ , and  $(f + I)(g + I) = fg + I$ . Note that if  $f \equiv g \pmod{I}$ , then  $f(\mathbf{s}) = g(\mathbf{s})$  for all  $\mathbf{s} \in \mathcal{V}_{\mathbb{R}}(I)$ .

We will say that a polynomial  $h \in \mathbb{R}[\mathbf{x}]$  is a sum of squares (*sos*) modulo  $I$  if there exist polynomials  $g_1, \dots, g_r \in \mathbb{R}[\mathbf{x}]$  such that  $h \equiv \sum_{i=1}^r g_i^2 \pmod{I}$ . If  $h$  is sos modulo  $I$  then we immediately have that  $h$  is nonnegative on  $\mathcal{V}_{\mathbb{R}}(I)$ . In practice, it is important to control the degree of the  $g_i$  in the sos representation of  $h$ , so we will say that  $h$

is  $k$ -sos mod  $I$  if  $g_1, \dots, g_r \in \mathbb{R}[\mathbf{x}]_k$ , where  $\mathbb{R}[\mathbf{x}]_k$  is the set of polynomials in  $\mathbb{R}[\mathbf{x}]$  of degree at most  $k$ . The set of polynomials that are  $k$ -sos mod  $I$ , considered as a subset of  $\mathbb{R}[\mathbf{x}]_{2k}/I$  will be denoted as  $\Sigma_k(I)$ .

**Definition 5.1.** Let  $I \subseteq \mathbb{R}[\mathbf{x}]$  be a polynomial ideal. We define the  $k$ -th *theta body* of  $I$  to be the set

$$\text{TH}_k(I) := \{\mathbf{p} \in \mathbb{R}^n : l(\mathbf{p}) \geq 0 \text{ for all } l \in \mathbb{R}[\mathbf{x}]_1 \text{ s.t. } l \text{ is } k\text{-sos mod } I\}.$$

Since, if  $l$  is sos mod  $I$  then  $l \geq 0$  on  $\mathcal{V}_{\mathbb{R}}(I)$ , and  $\text{TH}_k(I)$  is closed,  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I))) \subseteq \text{TH}_k(I)$ . Also,  $\text{TH}_k(I) \subseteq \text{TH}_{k-1}(I)$ , since as  $k$  increases, we are potentially intersecting more half-spaces. Thus, the theta bodies of  $I$  create a nested sequence of closed convex relaxations of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ .

We now present a related semidefinite programming relaxation of  $\mathcal{V}_{\mathbb{R}}(I)$  using the theory of moments.

For  $I \subseteq \mathbb{R}[\mathbf{x}]$  an ideal, let  $\mathcal{B} = \{f_0 + I, f_1 + I, \dots\}$  be a basis for the  $\mathbb{R}$ -vector space  $\mathbb{R}[\mathbf{x}]/I$ . We assume that the polynomials  $f_i$  representing the elements of  $\mathcal{B}$  are minimal degree representatives of their equivalence classes  $f_i + I$ . This makes the set  $\mathcal{B}_k := \{f_i + I \in \mathcal{B} : \deg(f_i) \leq k\}$  well-defined. In this chapter we will restrict ourselves to a special type of basis  $\mathcal{B}$ .

**Definition 5.2.** Let  $I \subseteq \mathbb{R}[\mathbf{x}]$  be an ideal. A basis  $\mathcal{B} = \{f_0 + I, f_1 + I, \dots\}$  of  $\mathbb{R}[\mathbf{x}]/I$  is a  $\theta$ -basis if it satisfies the following conditions:

1.  $f_0(\mathbf{x}) = 1$
2.  $f_i(\mathbf{x}) = x_i$ , for  $i = 1, \dots, n$
3.  $f_i$  is monomial for all  $i$
4. if  $\deg(f_i), \deg(f_j) \leq k$  then  $f_i f_j + I$  is in the real span of  $\mathcal{B}_{2k}$

We will also always assume that  $\mathcal{B}$  is ordered and that  $\deg(f_i) \leq \deg(f_{i+1})$ .

Using Gröbner bases theory, one can see that if  $\mathcal{V}_{\mathbb{R}}(I)$  is not contained in any proper affine space, then  $\mathbb{R}[\mathbf{x}]/I$  always has a  $\theta$ -basis. For instance, take the  $f_i$  in  $\mathcal{B}$  to be the *standard monomials* of an *initial ideal* of  $I$  with respect to some *total degree monomial ordering* on  $\mathbb{R}[\mathbf{x}]$  (see for example [2]). The methods we describe below work with non monomial bases of  $\mathbb{R}[\mathbf{x}]/I$  as explained in [5]. We restrict to a  $\theta$ -basis in this survey for ease of exposition and since the main applications we will discuss only need this type of basis.

Fix a  $\theta$ -basis  $\mathcal{B}$  of  $\mathbb{R}[\mathbf{x}]/I$  and define  $\mathbf{f}_k[\mathbf{x}]$  to be the column vector formed by all the elements of  $\mathcal{B}_k$  in order. Then  $\mathbf{f}_k[\mathbf{x}] \mathbf{f}_k[\mathbf{x}]^T$  is a square matrix indexed by  $\mathcal{B}_k$  with  $(i, j)$ -entry equal to  $f_i f_j + I$ . By hypothesis, the entries of  $\mathbf{f}_k[\mathbf{x}] \mathbf{f}_k[\mathbf{x}]^T$  lie in the  $\mathbb{R}$ -span of  $\mathcal{B}_{2k}$ . Let  $\{\lambda_{i,j}^l\}$  be the set of real numbers such that  $f_i f_j + I = \sum_{f_l + I \in \mathcal{B}_{2k}} \lambda_{i,j}^l f_l + I$ . We now linearize  $\mathbf{f}_k[\mathbf{x}] \mathbf{f}_k[\mathbf{x}]^T$  by replacing each element of  $\mathcal{B}_{2k}$  by a new variable.

**Definition 5.3.** Let  $I$ ,  $\mathcal{B}$  and  $\{\lambda_{i,j}^l\}$  be as above. Let  $\mathbf{y}$  be a real vector indexed by  $\mathcal{B}_{2k}$ . The  $k$ -th *combinatorial moment matrix*  $M_{\mathcal{B}_k}(\mathbf{y})$  of  $I$  is the real matrix indexed by  $\mathcal{B}_k$  whose  $(i, j)$ -entry is  $[M_{\mathcal{B}_k}(\mathbf{y})]_{i,j} = \sum_{f_l + I \in \mathcal{B}_{2k}} \lambda_{i,j}^l y_l$ .

*Example 5.1.* Let  $I \subseteq \mathbb{R}[x_1, x_2]$  be the ideal  $I = \langle x_1^2 - 2x_2 + x_1, x_1x_2 \rangle$ . Then a  $\theta$ -basis for  $I$  would be  $\mathcal{B} = \{1, x_1, x_2, x_2^2, x_2^3, x_2^4, \dots\}$ . Let us construct the matrix  $M_{\mathcal{B}_1}(\mathbf{y})$ . Consider the vector  $\mathbf{f}_1[\mathbf{x}] = (1 \ x_1 \ x_2)^T$ , then

$$\mathbf{f}_1[\mathbf{x}]\mathbf{f}_1[\mathbf{x}]^T = \begin{pmatrix} 1 & x_1 & x_2 \\ x_1 & x_1^2 & x_1x_2 \\ x_2 & x_1x_2 & x_2^2 \end{pmatrix} \equiv \begin{pmatrix} 1 & x_1 & x_2 \\ x_1 & 2x_2 - x_1 & 0 \\ x_1 & 0 & x_2^2 \end{pmatrix} \text{ mod } I.$$

We now linearize the resulting matrix using  $\mathbf{y} = (y_0, y_1, y_2, y_3)$ , where  $y_i$  indexes the  $i$ th element of  $\mathcal{B}$ , and get

$$M_{\mathcal{B}_1}(\mathbf{y}) = \begin{pmatrix} y_0 & y_1 & y_2 \\ y_1 & 2y_2 - y_1 & 0 \\ y_2 & 0 & y_3 \end{pmatrix}.$$

The matrix  $M_{\mathcal{B}_k}(\mathbf{y})$  will allow us to define a relaxation of  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$  that will essentially be the theta body  $\text{TH}_k(I)$ .

**Definition 5.4.** Let  $I \subseteq \mathbb{R}[\mathbf{x}]$  be an ideal and  $\mathcal{B}$  a  $\theta$ -basis of  $\mathbb{R}[\mathbf{x}]/I$ . Then

$$Q_{\mathcal{B}_k}(I) := \pi_{\mathbb{R}^n}(\{\mathbf{y} \in \mathbb{R}^{\mathcal{B}_{2k}} : y_0 = 1, M_{\mathcal{B}_k}(\mathbf{y}) \succeq 0\})$$

where  $\pi_{\mathbb{R}^n}$  is projection of  $\mathbf{y} \in \mathbb{R}^{\mathcal{B}_{2k}}$  to  $(y_1, \dots, y_n)$ , its coordinates indexed by  $x_1 + I, \dots, x_n + I$ .

The set  $Q_{\mathcal{B}_k}(I)$  is a relaxation of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ . Pick  $\mathbf{s} \in \mathcal{V}_{\mathbb{R}}(I)$  and define  $\mathbf{y}^s := (f_i(\mathbf{s}) : f_i + I \in \mathcal{B}_{2k})$ . Then  $\mathbf{y}^s(\mathbf{y}^s)^t = M_{\mathcal{B}_k}(\mathbf{y}^s)$ ,  $y_0^s = 1$  and  $\pi_{\mathbb{R}^n}(\mathbf{y}^s) = \mathbf{s}$ . We now show the connection between  $Q_{\mathcal{B}_k}(I)$  and  $\text{TH}_k(I)$ .

**Theorem 5.1.** For any ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$  and any  $\theta$ -basis  $\mathcal{B}$  of  $\mathbb{R}[\mathbf{x}]/I$ , we get  $\text{cl}(Q_{\mathcal{B}_k}(I)) \subseteq \text{TH}_k(I)$ .

*Proof.* We start with a general observation concerning  $k$ -sos polynomials. Suppose  $h \equiv \sum_{i=1}^r g_i^2 \text{ mod } I$  where  $g_i \in \mathbb{R}[\mathbf{x}]_k$ . Each  $g_i$  can be identified with a real row vector  $\hat{g}_i$  such that  $g_i(\mathbf{x}) \equiv \hat{g}_i \mathbf{f}_k[\mathbf{x}] \text{ mod } I$ , and so  $g_i^2 \equiv \mathbf{f}_k[\mathbf{x}]^T \hat{g}_i^T \hat{g}_i \mathbf{f}_k[\mathbf{x}]$ . Denoting by  $P_h$  the positive semidefinite matrix  $\sum_{i=1}^r \hat{g}_i^T \hat{g}_i$  we get  $h(\mathbf{x}) \equiv \mathbf{f}_k[\mathbf{x}]^T P_h \mathbf{f}_k[\mathbf{x}] \text{ mod } I$ . In general,  $P_h$  is not unique. Let  $\hat{h}$  be the real row vector such that  $h(\mathbf{x}) \equiv \hat{h} \mathbf{f}_{2k}[\mathbf{x}] \text{ mod } I$ . Then check that for any column vector  $\mathbf{y} \in \mathbb{R}^{\mathcal{B}_{2k}}$ ,  $\hat{h}\mathbf{y} = P_h \cdot M_{\mathcal{B}_k}(\mathbf{y})$ , where  $\cdot$  stands for the usual entry-wise inner product of matrices.

Suppose  $\mathbf{p} \in Q_{\mathcal{B}_k}(I)$ , and  $\mathbf{y} \in \mathbb{R}^{\mathcal{B}_{2k}}$  such that  $y_0 = 1$ ,  $M_{\mathcal{B}_k}(\mathbf{y}) \succeq 0$  and  $\pi_{\mathbb{R}^n}(\mathbf{y}) = \mathbf{p}$ . Since  $\text{TH}_k(I)$  is closed, we just have to show that for any  $h = a_0 + \sum_{i=1}^n a_i x_i \in \mathbb{R}[\mathbf{x}]_1$  that is  $k$ -sos modulo  $I$ ,  $h(\mathbf{p}) \geq 0$ . Since  $y_0 = 1$  and  $\pi_{\mathbb{R}^n}(\mathbf{y}) = \mathbf{p}$ ,  $h(\mathbf{p}) = a_0 + \sum_{i=1}^n a_i p_i = a_0 y_0 + \sum_{i=1}^n a_i y_i = \hat{h}\mathbf{y} = P_h \cdot M_{\mathcal{B}_k}(\mathbf{y}) \geq 0$  since  $P_h \succeq 0$  and  $M_{\mathcal{B}_k}(\mathbf{y}) \succeq 0$ .  $\square$

In the next subsection we will see that when  $I$  is a real radical ideal,  $\text{cl}(Q_{\mathcal{B}_k}(I))$  coincides with  $\text{TH}_k(I)$ .

The idea of computing approximations of the convex hull of a semialgebraic set in  $\mathbb{R}^n$  via the theory of moments and the dual theory of sums of squares polynomials is due to Lasserre [7–9] and Parrilo [20, 21]. In their original set up, the moment relaxations obtained are described via moment matrices that rely explicitly on the polynomials defining the semialgebraic set. In [8], the focus is on semialgebraic subsets of  $\{0, 1\}^n$  where the equations  $x_i^2 - x_i$  are used to simplify computations. This idea was generalized in [12] to arbitrary real algebraic varieties and studied in detail for zero-dimensional ideals. Laurent showed that the moment matrices needed in the approximations of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  could be computed modulo the ideal defining the variety. This greatly reduces the size of the matrices needed, and removes the dependence of the computation on the specific presentation of the ideal in terms of generators. The construction of the set  $Q_{\mathcal{B}_k}(I)$  is taken from [12]. Since an algebraic set is also a semialgebraic set (defined by equations), we could apply Lasserre's method to  $\mathcal{V}_{\mathbb{R}}(I)$  to get a sequence of approximations  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ . The results are essentially the same as theta bodies if the generators of  $I$  are picked carefully. However, by restricting ourselves to real varieties, instead of allowing inequalities, and concentrating on the sum of squares description of theta bodies, as opposed to the moment matrix approach, we can prove some interesting theoretical results that are not covered by the general theory for Lasserre relaxations. Many of the usual results for Lasserre relaxations rely on the existence of a non-empty interior for the semialgebraic set to be convexified which is never the case for a real variety, or compactness of the semialgebraic set which we do not want to impose.

### 5.2.2 Real Radical Ideals

Recall from the introduction that given an ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$  its real radical is the ideal

$$\sqrt{\mathbb{R}I} = \left\{ f \in \mathbb{R}[\mathbf{x}] : f^{2m} + \sum g_i^2 \in I, m \in \mathbb{N}, g_i \in \mathbb{R}[\mathbf{x}] \right\}.$$

The importance of this ideal arises from the Real Nullstellensatz.

**Theorem 5.2 (Bounded degree Real Nullstellensatz [16]).** *Let  $I \subseteq \mathbb{R}[\mathbf{x}]$  be an ideal. Then there exists a function  $\Gamma : \mathbb{N} \rightarrow \mathbb{N}$  such that, for all polynomials  $f \in \mathbb{R}[\mathbf{x}]$  of degree at most  $d$  that vanish on  $\mathcal{V}_{\mathbb{R}}(I)$ ,  $f^{2m} + \sum g_i^2 \in I$  for some polynomials  $g_i$  such that  $\deg(g_i)$  and  $m$  are all bounded above by  $\Gamma(d)$ . In particular  $I(\mathcal{V}_{\mathbb{R}}(I)) = \sqrt{\mathbb{R}I}$ .*

When  $I$  is a real radical ideal, the sums of squares approach and the moment approach for theta bodies of  $I$  coincide, and we get a stronger version of Theorem 5.1.

**Theorem 5.3.** *For any  $I \subseteq \mathbb{R}[\mathbf{x}]$  real radical and any  $\theta$ -basis  $\mathcal{B}$  of  $\mathbb{R}[\mathbf{x}]/I$ ,  $\text{cl}(Q_{\mathcal{B}_k}(I)) = \text{TH}_k(I)$ .*

*Proof.* By Theorem 5.1 we just have to show that  $\text{TH}_k(I) \subseteq \text{cl}(Q_{\mathcal{B}_k}(I))$ . By [22, Proposition 2.6], the set  $\Sigma_k(I)$  of elements of  $\mathbb{R}[\mathbf{x}]_{2k}/I$  that are  $k$ -sos modulo  $I$ , is closed when  $I$  is real radical. Let  $f \in \mathbb{R}[\mathbf{x}]_1$  be nonnegative on  $\text{cl}(Q_{\mathcal{B}_k}(I))$  and suppose  $f + I \notin \Sigma_k(I)$ . By the separation theorem, we can find  $\mathbf{y} \in \mathbb{R}^{\mathcal{B}_{2k}}$  such that  $\hat{f}\mathbf{y} < 0$  but  $\hat{g}\mathbf{y} \geq 0$  for all  $g + I \in \Sigma_k(I)$ , or equivalently,  $P_g \cdot M_{\mathcal{B}_k}(\mathbf{y}) \geq 0$ , where  $P_g$  is a positive semidefinite matrix associated to  $g$  as in the proof of Theorem 5.1. For any positive semidefinite matrix  $A$ , the polynomial  $h = f_k[\mathbf{x}]^t A f_k[\mathbf{x}]$  has  $A$  as a valid  $P_h$ , so  $P_g$  runs over all possible positive semidefinite matrices of size  $|\mathcal{B}_k|$ , and since the cone of positive semidefinite matrices of a fixed size is self-dual, we have  $M_{\mathcal{B}_k}(\mathbf{y}) \geq 0$ . Let  $r$  be any real number and consider  $g_r + I := (f + r)^2 + I \in \Sigma_k(I)$ . Then  $\hat{g}_r\mathbf{y} = \hat{f}^2\mathbf{y} + 2r\hat{f}\mathbf{y} + r^2y_0 \geq 0$ . Since  $\hat{f}\mathbf{y} < 0$  and  $r$  can be arbitrarily large,  $y_0$  is forced to be positive. So we can scale  $\mathbf{y}$  to have  $y_0 = 1$ , so that  $\pi_{\mathbb{R}^n}(\mathbf{y}) \in Q_{\mathcal{B}_k}(I)$ . By hypothesis we then have  $f(\pi_{\mathbb{R}^n}(\mathbf{y})) \geq 0$ , but by the linearity of  $f$ ,  $f(\pi_{\mathbb{R}^n}(\mathbf{y})) = \hat{f}\mathbf{y} < 0$  which is a contradiction, so  $f$  must be  $k$ -sos modulo  $I$ . This implies that any linear inequality valid for  $\text{cl}(Q_{\mathcal{B},k}(I))$  is valid for  $\text{TH}_k(I)$ , which proves  $\text{TH}_k(I) \subseteq \text{cl}(Q_{\mathcal{B},k}(I))$ .  $\square$

We now have two ways of looking at the relaxations  $\text{TH}_k(I)$  – one by a characterization of the linear inequalities that hold on them and the other by a characterization of the points in them. These two perspectives complement each other. The inequality version is useful to prove (or disprove) convergence of theta bodies to  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$  while the description via semidefinite programming is essential for practical computations. All the applications we consider use real radical ideals in which case the two descriptions of  $\text{TH}_k(I)$  coincide up to closure. In some cases, as we now show, the closure can be omitted.

**Theorem 5.4.** *Let  $I \subseteq \mathbb{R}[\mathbf{x}]$  be a real radical ideal and  $k$  be a positive integer. If there exists some linear polynomial  $g$  that is  $k$ -sos modulo  $I$  with a representing matrix  $P_g$  that is positive definite, then  $Q_{\mathcal{B}_k}(I)$  is closed and equals  $\text{TH}_k(I)$ .*

*Proof.* For this proof we will use a standard result from convex analysis: Let  $V$  and  $W$  be finite dimensional vector spaces,  $H \subseteq W$  be a cone and  $A : V \rightarrow W$  be a linear map such that  $A(V) \cap \text{int}(H) \neq \emptyset$ . Then  $(A^{-1}H)^* = A'(H^*)$  where  $A'$  is the adjoint operator to  $A$ . In particular,  $A'(H^*)$  is closed in  $V'$ , the dual vector space to  $V$ . The proof of this result follows, for example, from Corollary 3.3.13 in [1] by setting  $K = V$ .

Throughout the proof we will identify  $\mathbb{R}[\mathbf{x}]_l/I$ , for all  $l$ , with the space  $\mathbb{R}^{\mathcal{B}_l}$  by simply considering the coordinates in the basis  $\mathcal{B}_l$ . Consider the inclusion map  $A : \mathbb{R}^{\mathcal{B}_1} \cong \mathbb{R}^{n+1} \rightarrow \mathbb{R}^{\mathcal{B}_{2k}}$ , and let  $H$  be the cone in  $\mathbb{R}^{\mathcal{B}_{2k}}$  of polynomials that can be written as a sum of squares of polynomials of degree at most  $k$ . The interior of this cone is precisely the set of sums of squares  $g$  with a positive definite representing matrix  $P_g$ . Our hypothesis then simply states that  $A(\mathbb{R}^{\mathcal{B}_1}) \cap \text{int}(H) \neq \emptyset$  which implies by the above result that  $A'(H^*)$  is closed. Note that  $H^*$  is the set of elements  $\mathbf{y}$  in  $\mathbb{R}^{\mathcal{B}_{2k}}$  such that  $\hat{h}\mathbf{y}$  is nonnegative for all  $h \in H$  and this is the same as demanding  $P_h \cdot M_{\mathcal{B}_k}(\mathbf{y}) \geq 0$  for all positive semidefinite matrices  $P_h$ , which is equivalent to demanding that  $M_{\mathcal{B}_k}(\mathbf{y}) \geq 0$ . So  $A'(H^*)$  is just the set  $\pi_{\mathbb{R}^{n+1}}(\{\mathbf{y} \in \mathbb{R}^{\mathcal{B}_{2k}} : M_{\mathcal{B}_k}(\mathbf{y}) \geq 0\})$  and by intersecting it with the plane  $\{\mathbf{y} \in \mathbb{R}^{\mathcal{B}_{2k}} : y_0 = 1\}$  we get  $Q_{\mathcal{B}_k}(I)$  which is therefore closed.  $\square$

One very important case where the conditions of Theorem 5.4 holds is when  $I$  is the vanishing ideal of a set of 0/1 points. This is precisely the case of most interest in combinatorial optimization.

**Corollary 5.1.** *If  $S \subseteq \{0, 1\}^n$  and  $I = \mathcal{I}(S)$ , then  $Q_{\mathcal{B}_k}(I) = \text{TH}_k(I)$ .*

*Proof.* It is enough to show that there is a linear polynomial  $g \in \mathbb{R}[\mathbf{x}]$  such that  $g \equiv \mathbf{f}_k[\mathbf{x}]^T A \mathbf{f}_k[\mathbf{x}] \pmod{I}$  for a positive definite matrix  $A$  and some  $\theta$ -basis  $\mathcal{B}$  of  $\mathbb{R}[\mathbf{x}]/I$ .

Let  $A$  be the matrix

$$A = \begin{pmatrix} l+1 & \mathbf{c}^t \\ \mathbf{c} & D \end{pmatrix},$$

where  $l+1 = |\mathcal{B}_k|$ ,  $\mathbf{c} \in \mathbb{R}^l$  is the vector with all entries equal to  $-2$ , and  $D \in \mathbb{R}^{l \times l}$  is the diagonal matrix with all diagonal entries equal to  $4$ . This matrix is positive definite since  $D$  is positive definite and its Schur complement  $(l+1) - \mathbf{c}' D^{-1} \mathbf{c} = 1$  is positive.

Since  $x_i^2 \equiv x_i \pmod{I}$  for  $i = 1, \dots, n$  and  $\mathcal{B}$  is a monomial basis, for any  $f + I \in \mathcal{B}$ ,  $f \equiv f^2 \pmod{I}$ . Therefore, the constant (linear polynomial)  $l+1 \equiv \mathbf{f}_k[\mathbf{x}]^T A \mathbf{f}_k[\mathbf{x}] \pmod{I}$ .

□

The assumption that  $I$  is real radical seems very strong. However, we now establish a relationship between the theta bodies of an ideal and those of its real radical, showing that  $\sqrt[3]{I}$  determines the asymptotic behavior of  $\text{TH}_k(I)$ . We start by proving a small technical lemma.

**Lemma 5.1.** *Given an ideal  $I$  and a polynomial  $f$  of degree  $d$  such that  $-f^{2m} \in \Sigma_k(I)$  for some  $m, k \in \mathbb{N}$ , the polynomial  $f + \varepsilon \in \Sigma_{k+4dm}(I)$  for every  $\varepsilon > 0$ .*

*Proof.* First, note that for any  $l \geq m$  and any  $\xi > 0$  we have

$$f^l + \xi = \frac{1}{\xi} \left( \frac{f^l}{2} + \xi \right)^2 + \frac{1}{4\xi} (-f^{2m}) f^{2l-2m}$$

and so  $f^l + \xi$  is  $(dl+k)$ -sos modulo  $I$ . For  $\sigma > 0$ , define the polynomial  $p(x)$  to be the truncation of the Taylor series of  $\sqrt{\sigma^2 + 4\sigma x}$  at degree  $2m-1$  i.e.,

$$p(x) = \sum_{n=0}^{2m-1} (-1)^n \frac{(2n)!}{(n!)^2 (1-2n) \sigma^{n-1}} x^n.$$

When we square  $p(x)$  we get a polynomial whose terms of degree at most  $2m-1$  are exactly the first  $2m-1$  terms of  $\sigma^2 + 4\sigma x$ , and by checking the signs of each of the coefficients of  $p(x)$  we can see that the remaining terms of  $p(x)^2$  will be negative for odd powers and positive for even powers. Composing  $p$  with  $f$  we get

$$(p(f(\mathbf{x})))^2 = \sigma^2 + 4\sigma f(\mathbf{x}) + \sum_{i=0}^{m-1} a_i f(\mathbf{x})^{2m+2i} - \sum_{i=0}^{m-2} b_i f(\mathbf{x})^{2m+2i+1}$$

where the  $a_i$  and  $b_i$  are positive numbers. In particular

$$\sigma^2 + 4\sigma f(\mathbf{x}) = p(f(\mathbf{x}))^2 + \sum_{i=0}^{m-1} a_i f(\mathbf{x})^{2i} (-f(\mathbf{x})^{2m}) + \sum_{i=0}^{m-2} b_i f(\mathbf{x})^{2m+2i+1}.$$

On the right hand side of this equality the only term that is not immediately a sum of squares is the last one, but by the above remark, since  $2m+2i+1 > m$ , by adding an arbitrarily small positive number, it becomes  $(d(2m+2i+1)+k)$ -sos modulo  $I$ . By checking the degrees throughout the sum, one can see that for any  $\xi > 0$ ,  $\sigma^2 + 4\sigma f(\mathbf{x}) + \xi$  is  $(4dm+k)$ -sos modulo  $I$ . Since  $\sigma$  and  $\xi$  are arbitrary positive numbers we get the desired result.  $\square$

Lemma 5.1, together with the Real Nullstellensatz, gives us an important relationship between the theta body hierarchy of an ideal and that of its real radical.

**Theorem 5.5.** *Fix an ideal  $I$ . Then, there exists a function  $\Psi : \mathbb{N} \rightarrow \mathbb{N}$  such that  $\text{TH}_{\Psi(k)} \subseteq \text{TH}_k(\sqrt[k]{I})$  for all  $k$ .*

*Proof.* Fix some  $k$ , and let  $f(\mathbf{x})$  be a linear polynomial that is  $k$ -sos modulo  $\sqrt[k]{I}$ . This means that there exists some sum of squares  $s(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]_{2k}$  such that  $f - s \in \sqrt[k]{I}$ . Therefore, by the Real Nullstellensatz (Theorem 5.2),  $-(f - s)^{2m} \in \Sigma_l(I)$  for  $l, m \leq \Gamma(2k)$ , where  $\Gamma : \mathbb{N} \rightarrow \mathbb{N}$  depends only on the ideal  $I$ . By Lemma 5.1 it follows that  $f - s + \varepsilon$  is  $\Gamma(2k) + 8k\Gamma(2k)$ -sos modulo  $I$  for every  $\varepsilon > 0$ . Let  $\Psi(k) := \Gamma(2k) + 8k\Gamma(2k)$ . Then we have that  $f + \varepsilon$  is  $\Psi(k)$ -sos modulo  $I$  for all  $\varepsilon > 0$ . This means that for every  $\varepsilon > 0$ , the inequality  $f + \varepsilon \geq 0$  is valid on  $\text{TH}_{\Psi(k)}(I)$  for  $f$  linear and  $k$ -sos modulo  $\sqrt[k]{I}$ . Therefore,  $f \geq 0$  is also valid on  $\text{TH}_{\Psi(k)}(I)$ , and hence,  $\text{TH}_{\Psi(k)}(I) \subseteq \text{TH}_k(\sqrt[k]{I})$ .  $\square$

Note that the function  $\Psi$  whose existence we just proved, can be notoriously bad in practice, as it can be much higher than necessary. The best theoretical bounds on  $\Psi$  come from quantifier elimination and so increase very fast. For example, in [16], the best value for the function  $\Gamma$  in Theorem 5.2 is a tower of  $n+4$  exponentials. However, if we are only interested in convergence of the theta body sequence, as is often the case, Theorem 5.5 tells us that we might as well assume that our ideals are real radical.

### 5.3 Computing Theta Bodies

In this section we illustrate the computation of theta bodies on two examples. In the first example,  $\mathcal{V}_{\mathbb{R}}(I)$  is finite and hence  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  is a polytope, while in the second example  $\mathcal{V}_{\mathbb{R}}(I)$  is infinite. Convex approximations of polytopes via linear or semidefinite programming have received a great deal of attention in combinatorial optimization where the typical problem is to maximize a linear function  $\mathbf{c} \cdot \mathbf{x}$  as  $\mathbf{x}$  varies over the characteristic vectors  $\chi^T \in \{0, 1\}^n$  of some combinatorial objects  $T$ .

Since this discrete optimization problem is equivalent to the linear program in which one maximizes  $\mathbf{c} \cdot \mathbf{x}$  over all  $\mathbf{x} \in \text{conv}(\{\chi^T\})$  and  $\text{conv}(\{\chi^T\})$  is usually unavailable, one resorts to approximations of this polytope over which one can optimize in a reasonable way. A combinatorial optimization problem that has been tested heavily in this context is the *maximum stable set problem* in a graph which we use as our first example. In [17], Lovász constructed the *theta body of a graph* which was the first example of a semidefinite programming relaxation of a combinatorial optimization problem. The hierarchy of theta bodies for an arbitrary polynomial ideal are a natural generalization of Lovász's theta body for a graph, which explains their name. Our work on theta bodies was initiated by two problems that were posed by Lovász in [18, Problems 8.3 and 8.4] suggesting this generalization and its properties.

### 5.3.1 The Maximum Stable Set Problem

Let  $G = ([n], E)$  be an undirected graph with vertex set  $[n] = \{1, \dots, n\}$  and edge set  $E$ . A *stable set* in  $G$  is a set  $U \subseteq [n]$  such that for all  $i, j \in U$ ,  $\{i, j\} \notin E$ . The maximum stable set problem seeks the stable set of largest cardinality in  $G$ , the size of which is the *stability number* of  $G$ , denoted as  $\alpha(G)$ .

The maximum stable set problem can be modeled as follows. For each stable set  $U \subseteq [n]$ , let  $\chi^U \in \{0, 1\}^n$  be its *characteristic vector* defined as  $(\chi^U)_i = 1$  if  $i \in U$  and  $(\chi^U)_i = 0$  otherwise. Let  $S_G \subseteq \{0, 1\}^n$  be the set of characteristic vectors of all stable sets in  $G$ . Then  $\text{STAB}(G) := \text{conv}(S_G)$  is called the *stable set polytope* of  $G$  and the maximum stable set problem is, in theory, the linear program  $\max\{\sum_{i=1}^n x_i : \mathbf{x} \in \text{STAB}(G)\}$  with optimal value  $\alpha(G)$ . However,  $\text{STAB}(G)$  is not known apriori, and so one resorts to relaxations of it over which one can optimize  $\sum_{i=1}^n x_i$ .

In order to compute theta bodies for this example, we first need to view  $S_G$  as the real variety of an ideal. The natural ideal to take is  $\mathcal{I}(S_G)$ , the vanishing ideal of  $S_G$ . It can be checked that this real radical ideal is

$$I_G := \langle x_i^2 - x_i \forall i \in [n], x_i x_j \forall \{i, j\} \in E \rangle \subset \mathbb{R}[x_1, \dots, x_n].$$

For  $U \subseteq [n]$ , let  $\mathbf{x}^U := \prod_{i \in U} x_i$ . From the generators of  $I_G$  it follows that if  $f \in \mathbb{R}[\mathbf{x}]$ , then  $f \equiv g \pmod{I_G}$  where  $g$  is in the  $\mathbb{R}$ -span of the set of monomials  $\{\mathbf{x}^U : U \text{ is a stable set in } G\}$ . Check that

$$\mathcal{B} := \left\{ \mathbf{x}^U + I_G : U \text{ stable set in } G \right\}$$

is a  $\theta$ -basis of  $\mathbb{R}[\mathbf{x}]/I_G$  containing  $1 + I_G, x_1 + I_G, \dots, x_n + I_G$ . This implies that  $\mathcal{B}_k = \{\mathbf{x}^U + I_G : U \text{ stable set in } G, |U| \leq k\}$ , and for  $\mathbf{x}^{U_i} + I_G, \mathbf{x}^{U_j} + I_G \in \mathcal{B}_k$ , their product is

$\mathbf{x}^{U_i \cup U_j} + I_G$  which is  $0 + I_G$  if  $U_i \cup U_j$  is not a stable set in  $G$ . This product formula allows us to compute  $M_{\mathcal{B}_k}(\mathbf{y})$  where we index the element  $\mathbf{x}^U + I_G \in \mathcal{B}_k$  by the set  $U$ . Since  $S_G \subseteq \{0, 1\}^n$  and  $I_G = \mathcal{I}(S_G)$  is real radical, by Corollary 5.1, we have that

$$\text{TH}_k(I_G) = \left\{ \mathbf{y} \in \mathbb{R}^n : \begin{array}{l} \exists M \geq 0, M \in \mathbb{R}^{|\mathcal{B}_k| \times |\mathcal{B}_k|} \text{ such that} \\ M_{\emptyset\emptyset} = 1, \\ M_{\emptyset\{i\}} = M_{\{i\}\emptyset} = M_{\{i\}\{i\}} = y_i \\ M_{UU'} = 0 \text{ if } U \cup U' \text{ is not stable in } G \\ M_{UU'} = M_{WW'} \text{ if } U \cup U' = W \cup W' \end{array} \right\}.$$

In particular, indexing the one element stable sets by the vertices of  $G$ ,

$$\text{TH}_1(I_G) = \left\{ \mathbf{y} \in \mathbb{R}^n : \begin{array}{l} \exists M \geq 0, M \in \mathbb{R}^{(n+1) \times (n+1)} \text{ such that} \\ M_{00} = 1, \\ M_{0i} = M_{i0} = M_{ii} = y_i \forall i \in [n] \\ M_{ij} = 0 \forall \{i, j\} \in E \end{array} \right\}.$$

*Example 5.2.* Let  $G = (\{1, 2, 3, 4, 5\}, E)$  be a pentagon. Then

$$I_G = \langle x_i^2 - x_i \forall i = 1, \dots, 5, x_1x_2, x_2x_3, x_3x_4, x_4x_5, x_1x_5 \rangle$$

and

$$\mathcal{B} = \{1, x_1, x_2, x_3, x_4, x_5, x_1x_3, x_1x_4, x_2x_4, x_2x_5, x_3x_5\} + I_G.$$

Let  $\mathbf{y} \in \mathbb{R}^{11}$  be a vector whose coordinates are indexed by the elements of  $\mathcal{B}$  in the given order. Then

$$\text{TH}_1(I_G) = \left\{ \mathbf{y} \in \mathbb{R}^5 : \exists y_6, \dots, y_{10} \text{ s.t. } \begin{pmatrix} 1 & y_1 & y_2 & y_3 & y_4 & y_5 \\ y_1 & y_1 & 0 & y_6 & y_7 & 0 \\ y_2 & 0 & y_2 & 0 & y_8 & y_9 \\ y_3 & y_6 & 0 & y_3 & 0 & y_{10} \\ y_4 & y_7 & y_8 & 0 & y_4 & 0 \\ y_5 & 0 & y_9 & y_{10} & 0 & y_5 \end{pmatrix} \geq 0 \right\},$$

and  $\text{TH}_2(I_G) =$

$$\left\{ \mathbf{y} \in \mathbb{R}^5 : \exists y_6, \dots, y_{10} \text{ s.t. } \begin{pmatrix} 1 & y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 & y_9 & y_{10} \\ y_1 & y_1 & 0 & y_6 & y_7 & 0 & y_6 & y_7 & 0 & 0 & 0 \\ y_2 & 0 & y_2 & 0 & y_8 & y_9 & 0 & 0 & y_8 & y_9 & 0 \\ y_3 & y_6 & 0 & y_3 & 0 & y_{10} & y_6 & 0 & 0 & 0 & y_{10} \\ y_4 & y_7 & y_8 & 0 & y_4 & 0 & 0 & y_7 & y_8 & 0 & 0 \\ y_5 & 0 & y_9 & y_{10} & 0 & y_5 & 0 & 0 & 0 & y_9 & y_{10} \\ y_6 & y_6 & 0 & y_6 & 0 & 0 & y_6 & 0 & 0 & 0 & 0 \\ y_7 & y_7 & 0 & 0 & y_7 & 0 & 0 & y_7 & 0 & 0 & 0 \\ y_8 & 0 & y_8 & 0 & y_8 & 0 & 0 & 0 & y_8 & 0 & 0 \\ y_9 & 0 & y_9 & 0 & 0 & y_9 & 0 & 0 & 0 & y_9 & 0 \\ y_{10} & 0 & 0 & y_{10} & 0 & y_{10} & 0 & 0 & 0 & 0 & y_{10} \end{pmatrix} \succeq 0 \right\}.$$

It will follow from Proposition 5.1 below that

$$\text{STAB}(G) = \text{TH}_2(I_G) \subsetneq \text{TH}_1(I_G).$$

In general, it is a non-trivial task to decide whether two convex bodies coincide and thus to check whether a given theta body,  $\text{TH}_k(I)$ , equals  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$ . One technique is to show that all linear functions  $l(\mathbf{x})$  such that  $l(\mathbf{x}) \geq 0$  on  $\mathcal{V}_{\mathbb{R}}(I)$  are  $k$ -sos mod  $I$ . Two illustrations of this method appear shortly.

The first theta body of  $I_G$  is exactly the theta body,  $\text{TH}(G)$ , of  $G$  as defined by Lovász [19, Lemma 2.17]. The higher theta bodies  $\text{TH}_k(I_G)$  shown above give a nested sequence of convex relaxations of  $\text{STAB}(G)$  extending Lovász's  $\text{TH}(G)$ . The problem

$$\max \left\{ \sum_{i=1}^n x_i : \mathbf{x} \in \text{TH}(G) \right\}$$

can be solved to arbitrary precision in polynomial time in the size of  $G$  via semidefinite programming. The optimal value of this semidefinite program is called the *theta number* of  $G$  and provides an upper bound on  $\alpha(G)$ . See [6, Chap. 9] and [26] for more on the stable set problem and  $\text{TH}(G)$ . The body  $\text{TH}(G)$  was the first example of a semidefinite programming relaxation of a discrete optimization problem and snowballed the use of semidefinite programming in combinatorial optimization. See [14, 18] for surveys.

Recall that a graph  $G$  is *perfect* if and only if  $G$  has no induced odd cycles of length at least five, or their complements. Lovász showed that  $\text{STAB}(G) = \text{TH}(G)$  if and only if  $G$  is perfect. This equality shows that the maximum stable set problem

can be solved in polynomial time in the size of  $G$  when  $G$  is a perfect graph, and this geometric proof is the only one known for this complexity result. Since a pentagon is not perfect, it follows that for  $G$  a pentagon,  $\text{STAB}(G) \subsetneq \text{TH}_1(I_G)$ . We will see in Corollary 5.3 that when  $\mathcal{V}_{\mathbb{R}}(I)$  is finite, then there is some finite  $k$  for which  $\text{TH}_k(I) = \text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ . Since no monomial in the basis  $\mathcal{B}$  of  $\mathbb{R}[\mathbf{x}]/I_G$  has degree larger than  $\alpha(G)$ , for any  $G$ ,  $\text{STAB}(G) = \text{TH}_{\alpha(G)}(I_G)$  which proves that when  $G$  is a pentagon,  $\text{STAB}(G) = \text{TH}_2(I_G)$ . We now prove a second, general reason why  $\text{STAB}(G) = \text{TH}_2(I_G)$  when  $G$  is a pentagon.

A simple class of linear inequalities that are valid on  $\text{STAB}(G)$  are the *odd cycle inequalities*,  $\sum_{i \in C} x_i \leq \frac{|C|-1}{2}$ , where  $C$  is an odd cycle in  $G$ .

**Proposition 5.1.** *For any graph  $G$ , all odd cycle inequalities are valid on  $\text{TH}_2(I_G)$ .*

*Proof.* Let  $n = 2k + 1$  and  $C$  be an  $n$ -cycle. Then  $I_C = \langle x_i^2 - x_i, x_i x_{i+1} \mid i \in [n] \rangle$  where  $x_{n+1} = x_1$ . Therefore,  $(1 - x_i)^2 \equiv 1 - x_i$  and  $(1 - x_i - x_{i+1})^2 \equiv 1 - x_i - x_{i+1} \pmod{I_C}$ . This implies that,  $\pmod{I_C}$ ,

$$p_i^2 := ((1 - x_1)(1 - x_{2i} - x_{2i+1}))^2 \equiv p_i = 1 - x_1 - x_{2i} - x_{2i+1} + x_1 x_{2i} + x_1 x_{2i+1}.$$

Summing over  $i = 1, \dots, k$ , we get

$$\sum_{i=1}^k p_i^2 \equiv k - kx_1 - \sum_{i=2}^{2k+1} x_i + \sum_{i=3}^{2k} x_1 x_i \pmod{I_C}$$

since  $x_1 x_2$  and  $x_1 x_{2k+1}$  lie in  $I_C$ . Define  $g_i := x_1(1 - x_{2i+1} - x_{2i+2})$ . Then  $g_i^2 - g_i \in I_C$  and  $\pmod{I_C}$  we get that

$$\sum_{i=1}^{k-1} g_i^2 \equiv (k-1)x_1 - \sum_{i=3}^{2k} x_1 x_i, \text{ which implies } \sum_{i=1}^k p_i^2 + \sum_{i=1}^{k-1} g_i^2 \equiv k - \sum_{i=1}^{2k+1} x_i.$$

Thus,  $k - \sum_{i=1}^{2k+1} x_i$  is 2-sos mod  $I_C$ .

Now let  $G$  be any graph and  $C$  be an induced  $(2k+1)$ -cycle in  $G$ . Then since  $I_C \subseteq I_G$ ,  $k - \sum_{i=1}^{2k+1} x_i$  is 2-sos mod  $I_G$  which proves the result.  $\square$

**Corollary 5.2.** *If  $G$  is an odd cycle of length at least five,  $\text{STAB}(G) = \text{TH}_2(I_G)$ .*

*Proof.* It is known that the facet inequalities of  $\text{STAB}(G)$  for  $G$  an  $n = 2k + 1$ -cycle are:

$$x_i \geq 0 \quad \forall i \in [n], \quad 1 - x_i - x_{i+1} \geq 0 \quad \forall i \in [n], \quad \text{and} \quad k - \sum_{i=1}^n x_i \geq 0$$

(see for instance, [25, Corollary 65.12a]). Clearly,  $x_i$  is 2-sos mod  $I_G$  for all  $i \in [n]$ , and check that  $1 - x_i - x_j \equiv (1 - x_i - x_j)^2 \pmod{I_G}$  for all  $\{i, j\} \in E$ . By Proposition 5.1,  $k - \sum_{i=1}^n x_i$  is 2-sos mod  $I_G$ . If  $l(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]_1$  such that  $l(\mathbf{x}) \geq 0$  on  $\text{STAB}(G)$ , then  $l(\mathbf{x})$  is a nonnegative linear combination of the facet inequalities of  $\text{STAB}(G)$ , and hence  $l(\mathbf{x})$  is 2-sos mod  $I_G$ .  $\square$

Using the same method as in Proposition 5.1, one can show that the *odd antihole* and *odd wheel* inequalities [6, Chap. 9] that are valid for  $\text{STAB}(G)$  are also valid for  $\text{TH}_2(I_G)$ . Schoenebeck [24] has shown that there is no constant  $k$  such that  $\text{STAB}(G) = \text{TH}_k(I_G)$  for all graphs  $G$  (as expected, unless P=NP).

### 5.3.2 An Infinite Variety

Consider the cardioid with defining equation

$$(x_1^2 + x_2^2 + 2x_1)^2 = 4(x_1^2 + x_2^2).$$

This plane curve is the real variety of the ideal  $I = \langle h \rangle$  where

$$h = x_1^4 + 2x_1^2x_2^2 + x_2^4 + 4x_1^3 + 4x_1x_2^2 - 4x_2^2.$$

Points on this variety are parametrized by the angle  $\theta$  made by the line segment from the origin to the point, with the  $x_1$ -axis, by the equations:

$$x_1(\theta) = 2\cos\theta(1 - \cos\theta), \quad x_2(\theta) = 2\sin\theta(1 - \cos\theta).$$

It can be checked that the half space containing the origin defined by the tangent to the cardioid at  $(x_1(\theta), x_2(\theta))$  is

$$x_1 \frac{\sin 2\theta + \sin \theta}{2\cos^2 \theta + \cos \theta - 1} - x_2 + \frac{\sin 2\theta - 2\sin \theta}{2\cos^2 \theta + \cos \theta - 1} \geq 0.$$

We now compute theta bodies of this ideal. Since the defining ideal of the cardioid is generated by a quartic polynomial, no linear polynomial is a sum of squares of linear polynomials modulo this ideal. Therefore,  $\text{TH}_1(I) = \mathbb{R}^2$ . A  $\mathbb{R}$ -vector space basis  $\mathcal{B}$  of  $\mathbb{R}[x_1, x_2]/I$  is given by the monomials in  $x_1$  and  $x_2$  that are not divisible by  $x_1^4$ . In particular,

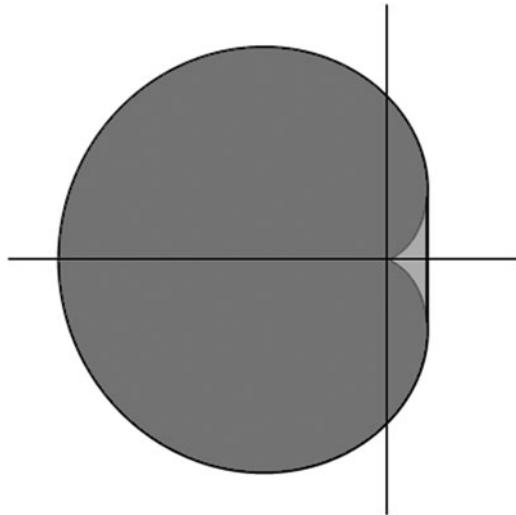
$$\mathcal{B}_4 = \left\{ 1, x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4 \right\} + I,$$

and suppose the coordinates of  $\mathbf{y} \in \mathbb{R}^{\mathcal{B}_4}$  are indexed as follows ( $y_0 = 1$ ):

$$\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|} \hline 1 & | & x_1 & | & x_2 & | & x_1^2 & | & x_1x_2 & | & x_2^2 & | & x_1^3 & | & x_1^2x_2 & | & x_1x_2^2 & | & x_2^3 & | & x_1^3x_2 & | & x_1^2x_2^2 & | & x_1x_2^3 & | & x_2^4 \\ \hline 1 & | & y_1 & | & y_2 & | & y_3 & | & y_4 & | & y_5 & | & y_6 & | & y_7 & | & y_8 & | & y_9 & | & y_{10} & | & y_{11} & | & y_{12} & | & y_{13} \\ \hline \end{array}$$

To compute  $M_{\mathcal{B}_2}(\mathbf{y})$  we need to express  $f_i f_j + I$  as a linear combination of elements in  $\mathcal{B}_4$  for every  $f_i + I, f_j + I \in \mathcal{B}_2$  and then linearize this linear combination using  $\mathbf{y}$ .

**Fig. 5.1**  $\text{TH}_2(I)$  compared with the cardioid



For example,  $x_1^4 + I = -2x_1^2x_2^2 - x_2^4 - 4x_1^3 - 4x_1x_2^2 + 4x_2^2 + I$  which linearizes to

$$T := -2y_{11} - y_{13} - 4y_6 - 4y_8 + 4y_5.$$

Doing all these computations shows that

$$\text{TH}_2(I) = \left\{ (y_1, y_2) : \exists \mathbf{y} \in \mathbb{R}^{13} \text{ s.t. } \begin{pmatrix} 1 & y_1 & y_2 & y_3 & y_4 & y_5 \\ y_1 & y_3 & y_4 & y_6 & y_7 & y_8 \\ y_2 & y_4 & y_5 & y_7 & y_8 & y_9 \\ y_3 & y_6 & y_7 & T & y_{10} & y_{11} \\ y_4 & y_7 & y_8 & y_{10} & y_{11} & y_{12} \\ y_5 & y_8 & y_9 & y_{11} & y_{12} & y_{13} \end{pmatrix} \geq 0 \right\}.$$

For a given  $\theta$  we can find the maximum  $t \in \mathbb{R}$  such that  $(t \cos(\theta), t \sin(\theta))$  is in  $\text{TH}_2(I)$ , using an SDP-solver. This point will be on the boundary of  $\text{TH}_2(I)$ , and if we vary  $\theta$ , we will trace that boundary. In Fig. 5.1 we show the result obtained by repeating this procedure over 720 equally spaced directions and solving this problem using SeDuMi 1.1R3 [28].

From the figure,  $\text{TH}_2(I)$  seems to match the convex hull of the cardioid, suggesting that  $\text{TH}_2(I) = \text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ . In fact, equality holds. To prove this equality, we would have to construct a sos representation modulo the ideal for each of the tangents to the cardioid. Independently of that, it is clear from the figure that  $\text{TH}_2(I)$  does a very good job of approximating this convex hull.

We now explain how one can also optimize a linear function over a theta body using the original sos definition of the body. For a vector  $\mathbf{c} \in \mathbb{R}^n$ , maximizing  $\mathbf{c} \cdot \mathbf{x}$

over  $\text{TH}_k(I)$  is the same as minimizing  $\lambda \in \mathbb{R}$  such that  $\lambda - \mathbf{c} \cdot \mathbf{x}$  is nonnegative on  $\text{TH}_k(I)$ . Under some mild assumptions such as compactness of  $\text{TH}_k(I)$  or  $I$  being real radical, this is precisely the same as minimizing  $\lambda \in \mathbb{R}$  such that  $\lambda - \mathbf{c} \cdot \mathbf{x}$  is  $k$ -sos modulo  $I$ .

Consider the previous example of the cardioid and let  $\mathbf{c} = (1, 1)$ . If we want to optimize in that direction over  $\text{TH}_2(I)$ , we need to minimize  $\lambda$  such that  $\ell_\lambda(x_1, x_2) := \lambda - x_1 - x_2$  is 2-sos modulo  $I$ . If  $\mathbf{f}_2(\mathbf{x}) := (1 \ x_1 \ x_2 \ x_1^2 \ x_1 x_2 \ x_2^2)^T$ , then  $\ell_\lambda(x_1, x_2)$  is 2-sos modulo  $I$  if and only if there exists a 6 by 6 positive semidefinite matrix  $A$  such that

$$\ell_\lambda(x_1, x_2) \equiv \mathbf{f}_2(\mathbf{x})^T A \mathbf{f}_2(\mathbf{x}) \pmod{I}.$$

By carrying out the multiplications and doing the simplifications using the ideal one gets that this is equivalent to finding  $A \succeq 0$  such that

$$\begin{aligned} \ell_\lambda(x_1, x_2) &= A_{11} + 2A_{12}x_1 + 2A_{13}x_2 + (2A_{14} + A_{22})x_1^2 \\ &\quad + (2A_{15} + 2A_{23})x_1x_2 + (2A_{16} + A_{33} + 4A_{44})x_2^2 \\ &\quad + (2A_{24} - 4A_{44})x_1^3 + (2A_{25} + 2A_{34})x_1^2x_2 \\ &\quad + (2A_{26} + 2A_{35} - 4A_{44})x_1x_2^2 + 2A_{36}x_2^3 + 2A_{45}x_1^3x_2 \\ &\quad + (2A_{46} + A_{55} - 2A_{44})x_1^2x_2^2 + 2A_{56}x_1x_2^3 + (A_{66} - A_{44})x_2^4 \end{aligned}$$

so our problem is minimizing  $\lambda$  such that there exists  $A \succeq 0$  verifying

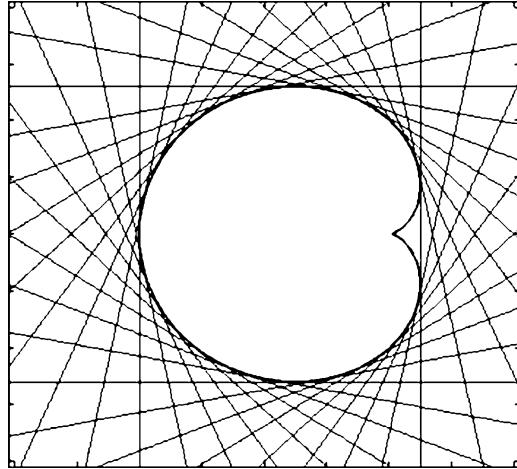
$$\begin{aligned} A_{11} &= \lambda; \\ A_{12} = A_{13} &= 1/2; 2A_{14} + A_{22} = 2A_{15} + 2A_{23} \\ &= 2A_{16} + A_{33} + 4A_{44} = 2A_{24} - 4A_{44} \\ &= 2A_{25} + 2A_{34} = 2A_{26} + 2A_{35} - 4A_{44} = A_{36} = A_{45} \\ &= 2A_{46} + A_{55} - 2A_{44} = A_{56} = A_{66} - A_{44} = 0. \end{aligned}$$

By taking  $\mathbf{c} = (\cos(\theta), \sin(\theta))$ , varying  $\theta$  and tracing all the optimal  $\ell_\lambda$  obtained one can get a contour of  $\text{TH}_2(I)$ . In Fig. 5.2 we can see all the  $\ell_\lambda$ 's obtained by repeating this process 32 times using SeDuMi.

## 5.4 Convergence and Exactness

In any hierarchy of relaxations, one crucial question is that of convergence: under what conditions does the sequence  $\text{TH}_k(I)$  approach  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$ ? Another important question is whether the limit is actually reached in a finite number of steps. In this section we will give a brief overview of the known answers to these questions.

**Fig. 5.2** Outer contour of  $\text{TH}_2(I)$



In general, convergence is not assured. In fact, for the ideal  $I = \langle x^3 - y^2 \rangle$ , whose real variety is a cusp, the closure of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  is the half plane defined by  $x \geq 0$ . However, all the theta bodies of  $I$  are just  $\mathbb{R}^2$ . This follows immediately from the fact that no linear polynomial can ever be written as a sum of squares modulo this ideal. Fortunately, many interesting ideals behave nicely with respect to the theta body hierarchy. In particular, the central result in this area tells us that compactness of  $\mathcal{V}_{\mathbb{R}}(I)$  implies convergence of the theta body hierarchy of  $I$ .

**Theorem 5.6.** *For any ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$  such that  $\mathcal{V}_{\mathbb{R}}(I)$  is compact,*

$$\bigcap_{k=0}^{\infty} \text{TH}_k(I) = \text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I))).$$

*Proof.* The inclusion  $\supseteq$  is clear. Suppose the other inclusion is false. Then pick a point  $\mathbf{p} \notin \text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$  such that  $\mathbf{p} \in \text{TH}_k(I)$  for all  $k$ . Then there exists a linear polynomial  $l(\mathbf{x})$  such that  $l(\mathbf{p}) < 0$ , and  $l(\mathbf{x}) > 0$  on  $\text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$  and therefore on  $\mathcal{V}_{\mathbb{R}}(I)$ . Let  $f_1, \dots, f_k$  be a set of generators for  $I$ . We can think of  $\mathcal{V}_{\mathbb{R}}(I)$  as the compact semialgebraic set  $S = \{\mathbf{x} \in \mathbb{R}^n : \pm f_i(\mathbf{x}) \geq 0\}$ . Then Schmüdgen's Positivstellensatz [23] applied to  $S$ , guarantees that  $l(\mathbf{x})$  has a sos representation modulo  $I$ . Hence,  $l(\mathbf{x}) \geq 0$  is valid for  $\text{TH}_t(I)$  for some  $t$  which contradicts that  $l(\mathbf{p}) < 0$ .  $\square$

In general, however, we are interested in finite convergence more than in asymptotic convergence, since in that case the theta bodies can give us a representation of the closure of the convex hull of the variety as the projection of a spectrahedron, an important theoretical problem on its own. We will say that an ideal  $I$  is **TH<sub>k</sub>-exact** if  $\text{TH}_k(I) = \text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$  and **TH-exact** if it is TH<sub>k</sub>-exact for some  $k$ . We first note that to study TH-exactness of an ideal depends only on the real radical of the ideal.

**Theorem 5.7.** *An ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$  is TH-exact if and only if  $\sqrt{\mathbb{R}I}$  is TH-exact.*

*Proof.* The “if” direction follows from Theorem 5.5, while the “only if” direction follows from the fact that  $I \subseteq \sqrt{\mathbb{R}I}$ .  $\square$

An important case in which TH-exactness holds is when  $\mathcal{V}_{\mathbb{R}}(I)$  is finite.

**Corollary 5.3.** *If  $I$  is an ideal such that  $\mathcal{V}_{\mathbb{R}}(I)$  is finite, then  $I$  is TH-exact.*

*Proof.* By Theorem 5.7 we can assume that  $I$  is real radical. If  $\mathcal{V}_{\mathbb{R}}(I) = \{\mathbf{p}_1, \dots, \mathbf{p}_m\}$ , then we can construct interpolators  $g_1, \dots, g_m \in \mathbb{R}[\mathbf{x}]$  such that  $g_i(\mathbf{p}_j) = 1$  if  $j = i$  and 0 otherwise. Assume that  $k$  is the highest degree of a  $g_i$ . Since  $g_i - g_i^2$  vanishes on  $\mathcal{V}_{\mathbb{R}}(I)$ ,  $g_i - g_i^2 \in \sqrt{\mathbb{R}I} = I$  and therefore,  $g_i$  is  $k$ -sos mod  $I$  for  $i = 1, \dots, m$ .

If  $f(\mathbf{x}) \geq 0$  on  $\mathcal{V}_{\mathbb{R}}(I)$ , then check that  $f(\mathbf{x}) \equiv \sum_{i=1}^m f(p_i)g_i(\mathbf{x}) \pmod{I}$ . Since the polynomial on the right hand side is a nonnegative combination of the  $g_i$ ’s which are  $k$ -sos mod  $I$ ,  $f$  is  $k$ -sos mod  $I$ . In particular, all the linear polynomials that are nonnegative on  $\mathcal{V}_{\mathbb{R}}(I)$  are  $k$ -sos mod  $I$  and so,  $\text{TH}_k(I) = \text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ .  $\square$

Under the stronger assumption that  $\mathcal{V}_{\mathbb{C}}(I)$  is finite, TH-exactness of  $I$  follows from a result of Parrilo (see [13] for a proof). The work in [10] implies that when  $\mathcal{V}_{\mathbb{R}}(I)$  is finite, the bodies  $Q_{\mathcal{B}_k}(I)$  converges in a finite number of steps, a slightly different result than the one in Corollary 5.3 – if  $I$  is real radical, then these results are the same, but since we do not assume that, we only have that  $Q_{\mathcal{B}_k}(I) \subseteq \text{TH}_k(I)$ . Finite varieties are of great importance in practice. For instance, they are precisely the feasible regions of the 0/1 integer programs.

To study TH-exactness in more detail, a better characterization is needed.

**Theorem 5.8.** *A real radical ideal  $I \subseteq \mathbb{R}[\mathbf{x}]$  is  $\text{TH}_k$ -exact if and only if all linear polynomials  $l$  that are non-negative in  $\mathcal{V}_{\mathbb{R}}(I)$  are  $k$ -sos modulo  $I$ .*

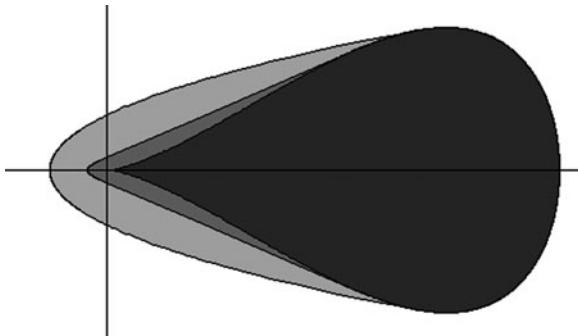
*Proof.* The “if” direction follows from the definition of theta bodies. The “only if” direction is a consequence of the proof of Theorem 5.3 which said that when  $I$  is real radical,  $\text{TH}_t(I) = \text{cl}(Q_{\mathcal{B}_t}(I))$  for all  $t$ . Recall that in the second part of that proof we showed that any linear inequality valid for  $\text{cl}(Q_{\mathcal{B}_t}(I))$  was  $t$ -sos mod  $I$ . Therefore, if  $\text{TH}_k(I) = \text{cl}(\text{conv}(\mathcal{V}_{\mathbb{R}}(I)))$ , then being nonnegative over  $\text{cl}(Q_{\mathcal{B}_k}(I)) = \text{TH}_k(I)$  is equivalent to being nonnegative over  $\mathcal{V}_{\mathbb{R}}(I)$  giving us the intended result.  $\square$

The last general result on exactness that we will present, taken from [4], is a negative one. Given a point  $\mathbf{p} \in \mathcal{V}_{\mathbb{R}}(I)$  we define the *tangent space* of  $\mathbf{p}$ ,  $T_{\mathbf{p}}(I)$ , to be the affine space passing through  $\mathbf{p}$  and orthogonal to the vector space spanned by the gradients of all polynomials that vanish on  $\mathcal{V}_{\mathbb{R}}(I)$ . We say that a point  $\mathbf{p}$  in  $\mathcal{V}_{\mathbb{R}}(I)$  is *convex-singular*, if it is on the boundary of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  and  $T_{\mathbf{p}}(I)$  is not tangent to  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  i.e.,  $T_{\mathbf{p}}(I)$  intersects the relative interior of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ .

**Theorem 5.9.** *An ideal with a convex-singularity is not TH-exact.*

*Proof.* Let  $I$  be an ideal and  $\mathbf{p}$  a convex-singular point of  $\mathcal{V}_{\mathbb{R}}(I)$ , and  $J = \mathcal{I}(\mathcal{V}_{\mathbb{R}}(I))$ . By Theorem 5.7, it is enough to show that  $J$  is not TH-exact. Let  $l(\mathbf{x})$  be a linear polynomial that is positive on the relative interior of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  and zero at  $\mathbf{p}$ ,

**Fig. 5.3** The second and third theta bodies of  $I = \langle x^4 - x^3 + y^2 \rangle$



which we can always find since  $\mathbf{p}$  is on the boundary of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$ . If  $J$  was  $\text{TH}_k$ -exact for some  $k$ , then since  $J$  is real radical, by Theorem 5.8 we would be able to write  $l(\mathbf{x}) = \sigma(\mathbf{x}) + g(\mathbf{x})$  where  $\sigma$  is a sum of squares and  $g \in J$ . Evaluating at  $\mathbf{p}$ , we see that  $\sigma(\mathbf{p}) = 0$ , which since  $\sigma$  is a sum of squares, implies that  $\nabla\sigma(\mathbf{p}) = 0$ . Therefore, we must have  $\nabla l = \nabla g(\mathbf{p})$ . Let  $\mathbf{q}$  be a point in the relative interior of  $\text{conv}(\mathcal{V}_{\mathbb{R}}(I))$  that is also in  $T_{\mathbf{p}}(I)$ . Then by the definition of  $T_{\mathbf{p}}(I)$ ,

$$l(\mathbf{q}) = (\mathbf{q} - \mathbf{p})\nabla l = (\mathbf{q} - \mathbf{p})\nabla g(\mathbf{p}) = \mathbf{0}$$

which contradicts our choice of  $l$ . Hence,  $I$  is not  $\text{TH}_k$ -exact.  $\square$

Even in the presence of convex-singular points, the theta bodies of the defining ideal can approximate the convex hull of the real variety arbitrarily well, but they will not converge in finitely many steps. An example of this is the ideal  $I = \langle x^4 - x^3 + y^2 \rangle$ . It has a compact real variety with a convex-singularity at the origin. Hence we have asymptotic convergence of the theta bodies, but not finite convergence. The first theta body of this ideal is  $\mathbb{R}^2$  and in Fig. 5.3 we can see that the next two theta bodies already closely approximate the convex hull of the variety.

In the remainder of this section we focus on a more restricted exactness question. Problem 8.3 in [18] motivates the question of which ideals in  $\mathbb{R}[\mathbf{x}]$  are  $\text{TH}_1$ -exact. Of particular interest are vanishing ideals of finite sets of points in  $\mathbb{R}^n$  and ideals arising in combinatorial optimization.

**Theorem 5.10.** *Let  $S$  be a finite set of points in  $\mathbb{R}^n$ , then  $I = \mathcal{I}(S)$  is  $\text{TH}_1$ -exact if and only if for each facet  $F$  of the polytope  $\text{conv}(S)$  there exists a hyperplane  $H$  parallel to  $F$  such that  $S \subseteq F \cup H$ .*

*Proof.* We can assume without loss of generality that  $\text{conv}(S)$  is full-dimensional, otherwise we could restrict ourselves to its affine hull. Assume  $I$  is  $\text{TH}_1$ -exact and let  $F$  be a facet of  $\text{conv}(S)$ . We can find a linear polynomial  $l$  such that  $l(\mathbf{p}) = 0$  for all  $\mathbf{p} \in F$  and  $l(\mathbf{p}) \geq 0$  for all  $\mathbf{p} \in S$ . By Theorem 5.8,  $l$  must be 1-sos modulo  $I$  which implies  $l \equiv \sum_{i=1}^r g_i^2 \pmod{I}$  for some linear polynomials  $g_i$ . In particular,  $l(\mathbf{p}) = \sum_{i=1}^r (g_i(\mathbf{p}))^2$  for all  $\mathbf{p} \in S$ , and since  $l$  vanishes in all points of  $S \cap F$ , so must the  $g_i$ . This implies that all the  $g_i$ 's vanish on the hyperplane  $\{\mathbf{x} \in \mathbb{R}^n : l(\mathbf{x}) = 0\}$ , since



**Fig. 5.4** The five 2-level polytopes in  $\mathbb{R}^3$  up to affine transformations

they are linear. This is equivalent to saying that every  $g_i$  is a scalar multiple of  $l$ , and therefore,  $l \equiv \lambda l^2 \bmod I$  for some nonnegative  $\lambda$ . Then  $l - \lambda l^2 \in I$  implies that  $S \subseteq \{\mathbf{x} : l(\mathbf{x}) - \lambda l(\mathbf{x})^2 = 0\}$  which is the union of the hyperplanes  $\{\mathbf{x} : l(\mathbf{x}) = 0\}$  and  $\{\mathbf{x} : l(\mathbf{x}) = \sqrt{1/\lambda}\}$ . So take  $H$  to be the second hyperplane.

Suppose now for each facet  $F$  of  $\text{conv}(S)$  there exists a hyperplane  $H$  parallel to  $F$  such that  $S \subseteq F \cup H$ . This implies that for any facet  $F$  and a fixed linear inequality  $l_F(\mathbf{x}) \geq 0$  that is valid on  $S$  and holds at equality on  $F$ ,  $l_F$  attains the same nonzero value at all points in  $S \setminus F$ . By scaling we can assume that value to be one, which implies that  $l_F(1 - l_F)$  is zero at all points in  $S$ , and hence,  $l_F(1 - l_F) \in I$ . But then  $l_F = l_F^2 + l_F(1 - l_F)$  is 1-sos mod  $I$ . By Farkas Lemma, any valid linear inequality for a polytope can be written as a nonnegative linear combination of the facet inequalities of the polytope. Therefore, any linear polynomial  $l$  that is nonnegative over  $S$  is a positive sum of 1-sos polynomials mod  $I$ , and hence,  $l$  is 1-sos mod  $I$ . Now using Theorem 5.8,  $I$  is  $\text{TH}_1$ -exact.  $\square$

We call a polytope  $P$  a  $k$ -level polytope if for any facet  $F$  of  $P$  and any supporting hyperplane  $H$  of  $F$ , there are  $k-1$  hyperplanes  $H_1, \dots, H_{k-1}$  all parallel to  $H$  such that the vertices of  $P$  are contained in  $H \cup H_1 \cup \dots \cup H_{k-1}$ . Theorem 5.10 states that  $\mathcal{I}(S)$  is  $\text{TH}_1$ -exact if and only if  $S$  is the set of vertices of a 2-level polytope. Polytopes with integer vertices that are 2-level are called *compressed polytopes* in the literature [27, 29].

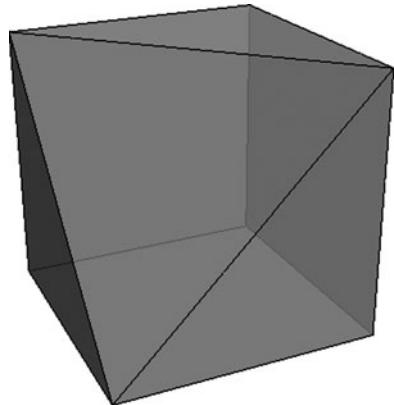
Examples of 2-level polytopes include simplices, cubes and cross-polytopes. In  $\mathbb{R}^3$ , up to affine transformations, there are only five different 2-level polytopes and they are shown in Fig. 5.4. An example of a polytope that is not 2-level is the truncated cube shown in Fig. 5.5. Three parallel translates of the hyperplane spanned by the slanted face are needed to contain all vertices of the polytope, and hence this is a 3-level polytope. Combinatorial properties of 2-level polytopes can be found in [5].

The result in Theorem 5.10 can be generalized to a (weak) sufficient criterion for  $\text{TH}_k$ -exactness.

**Theorem 5.11.** *If  $S \subseteq \mathbb{R}^n$  is the set of vertices of a  $(k+1)$ -level polytope then  $\mathcal{I}(S)$  is  $\text{TH}_k$ -exact.*

*Proof.* As before, we may assume that  $\text{conv}(S)$  is full-dimensional, and as observed in the proof of Theorem 5.10, it is enough to prove that for every facet  $F$  of  $\text{conv}(S)$ , if  $h_F$  is a linear polynomial that is zero on  $F$  and non-negative on  $S$  then  $h_F$  is k-sos modulo  $\mathcal{I}(S)$ . We can also assume that  $F$  is contained in the hyperplane

**Fig. 5.5** This polytope is 3-level but not 2-level



$\{\mathbf{x} \in \mathbb{R}^n : x_1 = 0\}$ , and that all points in  $S$  have nonnegative first coordinate, as all the properties we are interested in are invariant under translations.

Then, by scaling, we may assume  $h_F(\mathbf{x}) = x_1$ , and the  $(k+1)$ -level property tells us that there exist  $k$  positive values  $a_1, \dots, a_k$  such that all points in  $S$  have the first coordinate in the set  $\{0, a_1, \dots, a_k\}$ . Then we can construct a one variable Lagrange interpolator  $g$  of degree  $k$  such that  $g(0) = 0$  and  $g(a_i) = \sqrt{a_i}$  for  $i = 1, \dots, k$ . This will imply that  $h_F(\mathbf{x}) \equiv g(x_1)^2$  modulo  $\mathcal{I}(S)$  and so  $h_F$  is  $k$ -sos modulo  $\mathcal{I}(S)$ .  $\square$

This sufficient condition is very restrictive in general, and in fact can be arbitrarily bad.

*Example 5.3.* We saw in Corollary 5.2 that if  $G$  is a  $(2k+1)$ -cycle then the ideal  $I_G$  is  $\text{TH}_2$ -exact. However notice that we need  $k+1$  parallel translates of the hyperplane  $k = \sum x_i$  to cover all the vertices of  $\text{STAB}(G)$ , since the incidence vectors of the stable sets in  $G$  can have anywhere from 0 to  $k$  entries equal to one. Theorem 5.11 would only guarantee that  $I_G$  is  $\text{TH}_k$ -exact.

When  $\mathcal{V}_{\mathbb{R}}(I)$  is not finite,  $\text{TH}_1$ -exactness becomes harder to guarantee. A useful result in this direction is an alternative characterization of the first theta body of any ideal  $I$ . Given any ideal  $I$ , if we take all convex quadratic polynomials in  $I$ , and intersect the convex hulls of their zero sets, we obtain  $\text{TH}_1(I)$  exactly. This result, proved in [5, Theorem 5.4], can be used in some simple cases to both prove or disprove  $\text{TH}_1$ -exactness.

Many other questions on convergence remain open. For example, there is no known example of a smooth variety whose theta body hierarchy does not converge finitely, but there is also no reason to believe that such an example does not exist. A few examples of badly behaved smooth hypersurfaces, would be an important first step toward a better understanding of the theta body hierarchy for smooth varieties. Non-compact varieties have also not received much attention.

## 5.5 More Examples and Applications

### 5.5.1 The Maximum Cut Problem

In Sect. 5.3.1 we computed the theta body hierarchy for the maximum stable set problem. Another important problem in combinatorial optimization is the maximum cut (maxcut) problem. Given a graph  $G = ([n], E)$ , we say that  $C \subseteq E$  is a *cut* of  $G$ , if there exists some partition of the vertices into two sets  $V$  and  $W$  such that  $C$  is the set of edges between a vertex in  $V$  and a vertex in  $W$ . The maxcut problem is the problem of finding a cut in  $G$  of maximum cardinality. Theta bodies of the maxcut problem were studied in [3], and in this subsection we will present some of those results.

The model we use is similar to the one used in the stable set problem. The characteristic vector of a cut  $C$  of  $G$ , is the vector  $\chi_C \in \{-1, 1\}^E$  defined as  $\chi_C(e) = -1$  if  $e \in C$  and 1 otherwise. Let  $C_G$  be the collection of all characteristic vectors of cuts of  $G$ , and  $\mathcal{I}(C_G)$  the vanishing ideal of  $C_G$ . The maximum cardinality cut of  $G$  can be found, in principle, by optimizing  $\sum_{e \in E} x_e$  over  $C_G$ . However, this is a difficult problem and the size of the maxcut can be approximated by optimizing  $\sum_{e \in E} x_e$  over  $\text{TH}_k(\mathcal{I}(C_G))$ .

It is not hard to show that  $\mathcal{I}(C_G)$  is generated by the polynomials  $1 - x_e^2$  for all  $e \in E$ , together with the polynomials  $1 - \prod_{e \in K} x_e$  where  $K \subseteq E$  is a chordless cycle. Using this one can construct a  $\theta$ -basis for  $\mathcal{I}(C_G)$ , but to do that we need to introduce another combinatorial concept. Given an even set  $T \subseteq [n]$ , we call a subgraph  $H$  of  $G$  a  $T$ -join if the set of vertices of  $H$  with odd degree is precisely  $T$ . For example an  $\emptyset$ -join is a cycle, and the minimal  $\{s, t\}$ -join is the shortest path from  $s$  to  $t$ . It is clear that there exists a  $T$ -join if and only if  $T$  has an even number of nodes in each of the connected components of  $G$ . Define  $\mathcal{T}_G := \{T \subseteq [n] : \exists T\text{-join in } G\}$ , and for each  $T \in \mathcal{T}_G$ , choose  $H_T$  to be a  $T$ -join with a minimal number of edges. Define  $\mathcal{T}_k := \{T \subseteq \mathcal{T}_G : |H_T| \leq k\}$ , and note that  $\mathcal{T}_1 = \{\emptyset\} \cup E$ . Then one can show that  $\mathcal{B} = \{\prod_{e \in T} x_e + \mathcal{I}(C_G) : T \in \mathcal{T}_G\}$  is a  $\theta$ -basis for  $\mathcal{I}(C_G)$ , and we can therefore identify  $\mathcal{B}$  with  $\mathcal{T}_G$  and each  $\mathcal{B}_k$  with  $\mathcal{T}_k$ .

We can now give a description of the theta bodies of  $\mathcal{I}(C_G)$ .

$$\text{TH}_k(\mathcal{I}(C_G)) = \left\{ \mathbf{y} \in \mathbb{R}^E : \begin{array}{l} \exists M \geq 0, M \in \mathbb{R}^{|\mathcal{T}_k| \times |\mathcal{T}_k|} \text{ such that} \\ M_{e\emptyset} = y_e, \forall e \in E \\ M_{TT} = 1, \forall T \in \mathcal{T}_k \\ M_{T_1 T_1} = M_{T_3 T_4} \text{ if } T_1 \Delta T_2 = T_3 \Delta T_4 \end{array} \right\}.$$

In particular, since  $\mathcal{T}_1 = \{\emptyset\} \cup E$  we get

$$\text{TH}_1(\mathcal{I}(C_G)) = \left\{ \mathbf{y} \in \mathbb{R}^E : \begin{array}{l} \exists M \geq 0, M \in \mathbb{R}^{\{\emptyset\} \cup E \times \{\emptyset\} \cup E} \text{ such that} \\ M_{\emptyset\emptyset} = M_{ee} = 1, \forall e \in E \\ M_{\emptyset e} = M_{e\emptyset} = y_e \forall e \in E \\ M_{ef} = y_g \text{ if } \{e, f, g\} \text{ is a triangle in } G \\ M_{ef} = M_{gh} \text{ if } \{e, f, g, h\} \text{ is a square in } G \end{array} \right\}.$$

These relaxations are very closely related to some older hierarchies of relaxations for the maxcut problem. In particular they have a very interesting relation with the relaxation introduced in [11]. The theta body relaxation as it is, is not very strong, as shown by the following proposition.

**Proposition 5.2 ([3, Corollary 5.7]).** *Let  $G$  be the  $n$ -cycle. Then the smallest  $k$  for which  $\mathcal{I}(C_G)$  is  $\text{TH}_k$ -exact is  $k = \lceil n/4 \rceil$ .*

Convergence can be sped up for graphs with large cycles by taking a chordal closure of the graph, computing the theta body for this new graph, and then projecting to the space of edges of the original graph. If instead of the chordal closure one takes the complete graph over the same vertices, we essentially recover the hierarchy introduced in [11].

One can use the usual theta body hierarchy together with Theorem 5.10 to solve a question posed by Lovász in [18, Problem 8.4]. Motivated by the fact that a graph is perfect if and only if the stable set ideal  $I_G$  is  $\text{TH}_1$ -exact, Lovász asked to characterize the graphs that were “cut-perfect”, i.e., graphs  $G$  for which  $\mathcal{I}(C_G)$  is  $\text{TH}_1$ -exact. It turns out that such a “strong cut-perfect graph theorem” is not too hard to derive.

**Proposition 5.3 ([3, Corollary 4.12]).** *Given a graph  $G$ , the ideal  $\mathcal{I}(C_G)$  is  $\text{TH}_1$ -exact if and only if  $G$  had no  $K_5$  minor and no chordless circuit of length greater than or equal to five.*

### 5.5.2 Permutation Groups

In [15, Sect. 4], De Loera et al. study the behavior of the theta body hierarchy when applied to ideals associated to permutation groups. Let  $A \subseteq S_n$  be a subgroup of permutations and identify each element in  $A$  with a permutation matrix. We can then see these permutation matrices as vectors in  $\mathbb{R}^{n^2}$  and define  $I_A$  to be their vanishing ideal. One of the main goals in [15, Sect. 4] is to provide sufficient conditions for the  $\text{TH}_1$ -exactness of  $I_A$ .

A permutation group  $A$  is *permutation summable* if for all  $P_1, \dots, P_m \in A$  such that all entries of  $\sum P_i - I$  are nonnegative,  $\sum P_i - I$  is itself a sum of matrices in  $A$ . For example,  $S_n$  itself is permutation summable as a direct consequence

of Birkhoff's Theorem. We say that  $A$  is *strongly fixed-point free* if the only permutation in  $A$  that has fixed points is the identity. In terms of matrices this is equivalent to saying that an element of  $A$  is either the identity or has a zero diagonal. This trivially implies that all strongly fixed-point free groups are permutation summable.

**Proposition 5.4 ([15, Theorem 4.5]).** *If  $A \subseteq S_n$  is a permutation summable group, then  $I_A$  is  $\text{TH}_1$ -exact.*

A very interesting class of permutation groups is the automorphism groups of graphs. Given a graph  $G = ([n], E)$ ,  $\text{Aut}(G) \subseteq S_n$  is defined to be the group of vertex permutations  $\sigma$  such that  $\{i, j\} \in E$  implies  $\{\sigma(i), \sigma(j)\} \in E$ . The *automorphism polytope* of  $G$ , denoted as  $P_{\text{Aut}(G)}$ , is the convex hull of  $\text{Aut}(G)$ , where again we are identifying a permutation with its matrix representation. There has been some study of ways to relax this polytope. See [30] for instance. One such relaxation is the polytope  $P_G$  of all the points  $P \in \mathbb{R}^{n \times n}$  such that

$$\begin{aligned} PA_G = A_G P; \sum_{i=1}^n P_{i,j} &= 1, 1 \leq j \leq n; \\ \sum_{j=1}^n P_{i,j} &= 1, 1 \leq i \leq n; P_{i,j} \geq 0, 1 \leq i, j \leq n; \end{aligned}$$

where  $A_G$  is the  $n \times n$  adjacency matrix of  $G$ . If in the constraints of  $P_G$  we replace  $P_{i,j} \geq 0$  by  $P_{i,j} \in \{0, 1\}$ , then we get precisely  $\text{Aut}(G)$ . In fact,  $P_{\text{Aut}(G)}$  is the integer hull of  $P_G$  i.e., the convex hull of the integer points of  $P_G$ . If  $P_G = P_{\text{Aut}(G)}$  then  $G$  is said to be *compact*. It is not hard to show that  $\text{TH}_1(I_{\text{Aut}(G)}) \subseteq P_G$ , so the first theta body provides an approximation of  $P_{\text{Aut}(G)}$  that is at least as good as the linear approximation  $P_G$ , but one can actually say more.

**Proposition 5.5 ([15, Theorem 4.4]).** *The class of compact graphs is strictly included in the class of graphs with  $\text{TH}_1$ -exact automorphism ideal. In particular, let  $G_1, \dots, G_m$  be  $k$ -regular graphs (all vertices have degree  $k$ ) that are compact and  $G$  their disjoint union. Then  $I_{\text{Aut}(G)}$  is  $\text{TH}_1$ -exact, but  $G$  is compact if and only if  $G_1 \cong \dots \cong G_m$ .*

Finally note that while theta bodies of automorphism groups of graphs have interesting theoretical properties, computing with them tends to be quite hard, as there is no easy general way to obtain a good  $\theta$ -basis for this problem. In fact, knowing if a graph has a non-trivial automorphism, the *graph automorphism problem*, is equivalent to knowing if a  $\theta$ -basis for  $I_{\text{Aut}(G)}$  has more than one element. Determining the complexity class of this problem is a major open question.

**Acknowledgements** Both authors were partially supported by the NSF Focused Research Group grant DMS-0757371. J. Gouveia was also partially supported by Fundação para a Ciência e Tecnologia and R.R. Thomas by the Robert R. and Elaine K. Phelps Endowed Professorship.

## References

1. Borwein, J.M., Lewis, A.S.: Convex analysis and nonlinear optimization. Theory and examples. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC, 3. Springer, New York, second edition (2006)
2. Cox, D., Little, J., O’Shea, D.: Ideals, Varieties and Algorithms. Springer-Verlag, New York (1992)
3. Gouveia, J., Laurent, M., Parrilo, P., Thomas, R.: A new hierarchy of semidefinite programming relaxations for cycles in binary matroids and cuts in graphs. *Math. Program.*, Ser. A DOI 10.1007/s10107-010-0425-z
4. Gouveia, J., Netzer, T.: Positive polynomials and projections of spectrahedra. *SIAM J. Optim.*, to appear
5. Gouveia, J., Parrilo, P.A., Thomas, R.R.: Theta bodies for polynomial ideals. *SIAM Journal on Optimization* **20**(4), 2097–2118 (2010)
6. Grötschel, M., Lovász, L., Schrijver, A.: Geometric algorithms and combinatorial optimization, Vol. 2 of Algorithms and Combinatorics. Springer-Verlag, Berlin, second edition (1993)
7. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. *SIAM J. Optim.* **11**(3), 796–817 (2001)
8. Lasserre, J.B.: An explicit equivalent positive semidefinite program for nonlinear 0-1 programs. *SIAM J. Optim.* **12**(3), 756–769 (2002)
9. Lasserre, J.B.: Convex sets with semidefinite representation. *Math. Program.* **120**, 457–477 (2009)
10. Lasserre, J.B., Laurent, M., Rostalski, P.: Semidefinite characterization and computation of zero-dimensional real radical ideals. *Found. Comput. Math.* **8**(5), 607–647 (2008)
11. Laurent, M.: Semidefinite relaxations for max-cut. In: The sharpest cut, MPS/SIAM Ser. Optim., pp. 257–290. SIAM, Philadelphia, PA (2004)
12. Laurent, M.: Semidefinite representations for finite varieties. *Math. Program.* **109**(1, Ser. A), 1–26 (2007)
13. Laurent, M.: Sums of squares, moment matrices and optimization over polynomials. In: Putinar, M., Sullivant, S. (eds.) Emerging Applications of Algebraic Geometry, IMA Volumes in Mathematics and its Applications, vol. 149, pp. 157–270. Springer (2009)
14. Laurent, M., Rendl, F.: Semidefinite programming and integer programming. In: Aardal, K., Nemhauser, G., Weismantel, R. (eds.) Handbook on Discrete Optimization, pp. 393–514. Elsevier B.V. (2005)
15. De Loera, J., Hillar, C., Malkin, P.N., Omar, M.: Recognizing graph theoretic properties with polynomial ideals. *Electron. J. Combinator.*, R114, 17(1), 2010
16. Lombardi, H.: Une borne sur les degrés pour le théorème des zéros réel effectif. In: Real algebraic geometry, Lecture Notes in Math., vol. 1524, pp. 323–345. Springer, Berlin (1992)
17. Lovász, L.: On the Shannon capacity of a graph. *IEEE Trans. Inform. Theory* **25**(1), 1–7 (1979)
18. Lovász, L.: Semidefinite programs and combinatorial optimization. In: Recent advances in algorithms and combinatorics, CMS Books Math./Ouvrages Math. SMC, vol. 11, pp. 137–194. Springer, New York (2003)
19. Lovász, L., Schrijver, A.: Cones of matrices and set-functions and 0-1 optimization. *SIAM J. Optim.* **1**(2), 166–190 (1991)
20. Parrilo, P.A.: Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. Ph.D. thesis, California Institute of Technology (2000)
21. Parrilo, P.A.: Semidefinite programming relaxations for semialgebraic problems. *Math. Prog.* **96**(2, Ser. B), 293–320 (2003)
22. Powers, V., Scheiderer, C.: The moment problem for non-compact semialgebraic sets. *Adv. Geom.* **1**(1), 71–88 (2001)
23. Schmüdgen, K.: The  $K$ -moment problem for compact semi-algebraic sets. *Math. Ann.* **289**(2), 203–206 (1991)

24. Schoenebeck, G.: Linear level Lasserre lower bounds for certain k-csps. In: FOCS, IEEE Computer Society, pp. 593–602 (2008)
25. Schrijver, A.: Combinatorial optimization. Polyhedra and efficiency. Algorithms and Combinatorics, Vol. 24B, Springer-Verlag, Berlin (2003)
26. Shepherd, F.B.: The theta body and imperfection. In: Perfect graphs, Wiley-Intersci. Ser. Discrete Math. Optim., pp. 261–291. Wiley, Chichester (2001)
27. Stanley, R.P.: Decompositions of rational convex polytopes. Combinatorial mathematics, optimal designs and their applications (Proc. Sympos. Combin. Math. and Optimal Design, Colorado State Univ., Fort Collins, Colo., 1978). Ann. Discrete Math. **6**, 333–342 (1980)
28. Sturm, J.F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. Optim. Methods Softw. **11/12**(1-4), 625–653 (1999)
29. Sullivant, S.: Compressed polytopes and statistical disclosure limitation. Tohoku Math. J. (2) **58**(3), 433–445 (2006)
30. Tinhofer, G.: Graph Isomorphism and Theorems of Birkhoff Type. Computing **36**, 285–300 (1986)

# Chapter 6

## Convex Relaxations and Integrality Gaps

Eden Chlamtac and Madhur Tulsiani

### 6.1 Introduction

Convex relaxations are one of the most powerful techniques for designing polynomial time approximation algorithms for NP-hard optimization problems such as Chromatic Number, MAX-CUT, Minimum Vertex Cover etc. Approximation algorithms for these problems are developed by formulating the problem at hand as an integer program. One then relaxes the integer program to a convex program which can be solved in polynomial time, such as a linear program (LP) or semidefinite program (SDP). A solution to the combinatorial problem is then obtained by designing a (possibly randomized) polynomial-time algorithm to convert the solution of such a convex relaxation, to an integer solution for the combinatorial problem, often referred to as “rounding”.

If we are dealing with (say) a maximization problem for which the true combinatorial optimum is  $\text{OPT}$ , then the convex relaxation will achieve a value  $\text{FRAC}$  which is at least as large as  $\text{OPT}$  (as the integer solution is also a feasible solution to the convex program). The rounding algorithm then uses the solution of the convex relaxation with objective value  $\text{FRAC}$  to produce an integer solution with (possibly suboptimal) value  $\text{ROUND}$ . The analysis of the algorithm then boils down to a comparison of these three quantities which satisfy  $\text{ROUND} \leq \text{OPT} \leq \text{FRAC}$ . The inequalities are reversed for a minimization problem.

If one just thinks of the combinatorial problem as a question of finding the (say) maximum *value* of the objective (e.g. the *size* of the maximum cut in a graph),

---

E. Chlamtac (✉)

Tel Aviv University, Tel Aviv, Israel

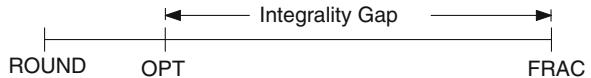
e-mail: [chlamtac@post.tau.ac.il](mailto:chlamtac@post.tau.ac.il)

M. Tulsiani

Toyota Technological Institute at Chicago, Chicago, IL, USA

e-mail: [madhurt@ttic.edu](mailto:madhurt@ttic.edu)

**Fig. 6.1** The integrality gap for a maximization problem



then the rounding algorithm is not needed and the quality of approximation is measured by the ratio  $FRAC/ROUND$ . If instead, the question is to *search* for an optimum integer solution (e.g. a cut of maximum value), then one is interested in the ratio  $OPT/ROUND$ . However, the analyses of most approximation algorithms actually require proving an upper bound on  $FRAC/ROUND$  (which in turn bounds  $OPT/ROUND$ ), simply because  $OPT$  is not known for an arbitrary instance of the problem! Thus a lower bound on the ratio  $FRAC/OPT$  not only gives a lower bound on the quality of approximation for the objective value, but also on the performance of rounding algorithms (if the algorithms are analyzed by comparing  $FRAC$  and  $ROUND$ ). This ratio is called the *integrality gap* of the program. Figure 6.1 shows the relationship between these quantities.

For a (maximization) problem, the integrality gap of a program is defined as the supremum of the ratio  $FRAC/OPT$  over all instances of the problem. For a minimization, we take it to be the supremum of the inverse ratio. Note that the integrality gap is always at least 1 and a large gap indicates a poor approximation ratio. Conversely, the analysis of an approximation algorithm bounding  $FRAC/ROUND$  in turn gives an *upper bound* on the integrality gap. In cases when the integrality gap is infinite, we express it as a function of the size of the instance, in which case it is defined as the maximum of the relevant ratio over all instances of the given size.

## 6.2 Hierarchies of Convex Relaxations

Convex relaxations for various combinatorial problems can be strengthened by including additional constraints which are satisfied by an integer solution. This process of generating stronger relaxations by adding larger (but still, local) constraints is captured by various hierarchies of convex relaxations such as the ones defined by Lovász and Schrijver [47], Sherali and Adams [55] and Lasserre [42]. Starting from a basic relaxation, these hierarchies define various *levels* of convex relaxations for a problem, with the relaxations at a higher level being more powerful than the relaxations at lower levels.

These hierarchies are known to capture the convex relaxations used in the best available algorithms for many problems, such as the SDP relaxation for Sparsest Cut by Arora et al. [7] and the  $\vartheta$ -function of Lovász for Maximum Independent Set [46], within a constant number of levels. It is also known that for an integer program with  $n$  variables taking values in  $\{0, 1\}$ , the convex program obtained by  $n$  levels of any of the above hierarchies has integrality gap 1, i.e., it gives the exact solution. However, solving the program obtained by  $t$  levels of these hierarchies takes time  $n^{O(t)}$  which is exponential in  $n$  for  $t = \Omega(n)$ .

<u>LP relaxation</u>	<u>SDP relaxation</u>
$\text{maximize} \quad \sum_{i \in V} x_i$ $\text{subject to} \quad x_i + x_j \leq 1 \quad \forall (i, j) \in E$ $x_i \in [0, 1]$	$\text{maximize} \quad \sum_{i \in V} \ u_i\ ^2$ $\text{subject to} \quad \langle u_i, u_j \rangle = 0 \quad \forall (i, j) \in E$ $\langle u_i, u_0 \rangle = \ u_i\ ^2 \quad \forall i \in V$ $\ u_0\  = 1$

**Fig. 6.2** LP and SDP relaxations for Maximum Independent Set

The interesting question is then to characterize the problems for which the  $t$ th level of these hierarchies yields a better approximation, for a small  $t$ . On the other hand, a lower bound showing that the integrality gap of the program obtained after many (say even  $\mathcal{O}(n)$ ) levels of a hierarchy remains large, is a strong lower bound against a class of algorithms capturing most known ones.

We describe below each of these hierarchies.<sup>1</sup> We shall use the example of Maximum Independent Set throughout this chapter to illustrate the differences in the programs obtained by the various hierarchies. An excellent comparison of all the three hierarchies mentioned above is also available in [43]. The basic LP and SDP relaxations for Maximum Independent Set (also known as Maximum Stable Set) are given in Fig. 6.2. We follow the convention used in the study of approximation algorithms of writing the SDP directly in terms of inner products of vectors in the Cholesky decomposition<sup>2</sup> of the PSD matrix of variables.

### 6.2.1 The Lovász–Schrijver Hierarchies

Lovász and Schrijver [47] describe two versions of a “lift-and-project” method. This can be thought of as an operator which when applied to a convex programming relaxation  $\mathcal{P}$  of a 0/1 integer linear program, produces a tighter relaxation. A weaker version of the method, denoted LS, adds auxiliary variables and linear inequalities, and the projection of the new relaxation on the original variables is denoted by  $N(\mathcal{P})$ ; a stronger version, denoted LS+, adds semidefinite programming constraints as well, and the projection on the original variables is denoted by  $N_+(\mathcal{P})$ .

Starting from a basic relaxation and iteratively applying the operator  $N$  ( $N_+$ ) one gets higher and higher levels (which are called *rounds* for the Lovász–Schrijver

<sup>1</sup>Some of these hierarchies can be defined in a more general context. However, we shall limit our discussion to relaxations of 0/1 (or  $-1/1$ ) integer programs.

<sup>2</sup>The Cholesky decomposition of an  $n \times n$  PSD matrix  $X$  is a collection of vectors  $u_1, \dots, u_n$  satisfying for all  $i, j$ ,  $X_{ij} = \langle u_i, u_j \rangle$ .

hierarchies due to their iterative nature) of the LS (LS+) hierarchy. Thus, the relaxation obtained by  $r$  rounds of the hierarchy is given by  $N(\cdots N(\mathcal{P}) \cdots)$  where the operator is applied  $t$  times. We denote it as  $N^t(\mathcal{P})$ .

Lovász and Schrijver also prove that if we start from a linear programming relaxation of a 0/1 integer program with  $n$  variables, then  $n$  applications of the LS procedures are sufficient to obtain a tight relaxation where the only feasible solutions are convex combinations of integral solutions. If one starts from a linear program with  $\text{poly}(n)$  inequalities, then it is possible to optimize over the set of solutions defined by  $t$  rounds of LS or LS+ in  $n^{O(t)}$  time.

To describe these hierarchies it will be more convenient to work with *convex cones* rather than arbitrary convex subsets of  $[0, 1]^n$ . Specifically, if we are interested in a convex set  $\mathcal{P} \subseteq [0, 1]^n$  (which might be the feasible region of our starting convex relaxation), we first convert it into the cone  $\text{cone}(\mathcal{P}) \subseteq \mathbb{R}^{n+1}$  defined as the set of all vectors  $(\lambda, \lambda y_1, \dots, \lambda y_N)$  such that  $\lambda \geq 0$  and  $(y_1, \dots, y_n) \in \mathcal{P}$ . For example, in the “cone” linear programming relaxation of the Maximum Independent Set problem  $(y_0, y_1, \dots, y_n)$  is in the feasible region (denoted by  $\text{cone}(\text{IS}(G))$ ) if and only if

$$\begin{aligned} y_i + y_j &\leq y_0 \quad \forall (i, j) \in E \\ 0 \leq y_i &\leq y_0 \quad \forall i \in V \\ y_0 &\geq 0 \end{aligned} \tag{\text{cone}(\text{IS}(G))}$$

We would now like to “tighten” the relaxation by adding inequalities (on the solution obtained after scaling to get  $y_0 = 1$ ) that are valid for 0/1 solutions but that are violated by other solutions. Ideally, we would like to say that a solution  $(1, y_1, \dots, y_n)$  must satisfy the conditions  $y_i^2 = y_i$ , because such a condition is satisfied only by 0/1 solutions. Equivalently, we could introduce  $n^2$  new variables  $Y_{i,j}$  and add the conditions (i)  $Y_{i,j} = y_i \cdot y_j$  and (ii)  $Y_{i,i} = y_i$ . Unfortunately, condition (i) is neither linear nor convex, and so we will instead “approximate” condition (i) by enforcing a set of linear conditions that are implied by (but not equivalent to) (i). This is formalized in the definition below.

**Definition 6.1.** For a cone  $K \subseteq \mathbb{R}^d$  we define the set  $N(K)$  (also a cone in  $\mathbb{R}^d$ ) as follows: a vector  $\mathbf{y} = (y_0, \dots, y_{d-1}) \in \mathbb{R}^d$  is in  $N(K)$  if and only if there is a matrix  $Y \in \mathbb{R}^{d \times d}$  such that

1.  $Y$  is symmetric
2. For every  $i \in \{0, 1, \dots, d-1\}$ ,  $Y_{0,i} = Y_{i,0} = y_i$
3. Each row  $Y_i$  is an element of  $K$
4. Each vector  $Y_0 - Y_i$  is an element of  $K$

In such a case,  $Y$  is called the *protection matrix* of  $\mathbf{y}$ . If, in addition,  $Y$  is *positive semidefinite*, then  $\mathbf{y} \in N_+(K)$ . We define  $N^0(K)$  and  $N_+^0(K)$  as  $K$ , and  $N^t(K)$  (respectively,  $N_+^t(K)$ ) as  $N(N^{t-1}(K))$  (respectively,  $N_+(N_+^{t-1}(K))$ ). When  $K = \text{cone}(\mathcal{P})$  for  $\mathcal{P} \subseteq \mathbb{R}^{d-1}$ , we denote as  $N^t(\mathcal{P})$  the set  $\{\mathbf{y} \in \mathbb{R}^{d-1} \mid (1, \mathbf{y}) \in N^t(\text{cone}(\mathcal{P}))\}$ , and similarly for  $N_+^t(\mathcal{P})$ .

Let us see that these operators are in fact relaxations for condition (i) above. Indeed, if  $\mathbf{y} = (1, y_1, \dots, y_{d-1}) \in \{0, 1\}^d$ , then we can set  $Y_{i,j} = y_i \cdot y_j$ . Such a matrix  $Y$  is clearly positive semidefinite, and it satisfies  $Y_{i,i} = y_i^2 = y_i$  if the  $y_i$  are in  $\{0, 1\}$ . Consider now a row  $Y_i$  of  $Y$ , that is, the vector  $\mathbf{r}$  such that  $r_j := Y_{i,j} = y_i \cdot y_j$ . Then, either  $y_i = 0$ , in which case  $\mathbf{r} = (0, \dots, 0)$  is in every cone, or  $y_i = 1$ , and  $\mathbf{r} = \mathbf{y}$ . Similarly, if we consider  $r_j := Y_{0,j} - Y_{i,j} = (1 - y_i) \cdot y_j$  we find that  $\mathbf{r}$  either equals the all-zero vector or it equals  $\mathbf{y}$ . This shows that if  $\mathbf{y} = (1, y_1, \dots, y_{d-1}) \in \{0, 1\}^d$  and  $\mathbf{y} \in K$ , then also  $\mathbf{y} \in N_+^t(K)$  for every  $t$ . Hence, if  $K \cap \{y_0 = 1\}$  defines a relaxation of the integral problem, so does  $N_+^t(K) \cap \{y_0 = 1\}$ , and hence also  $N^t(K) \cap \{y_0 = 1\}$ .

For a graph  $G$ , the relaxation of the Maximum Independent Set problem resulting from  $t$  rounds of LS+ is the result of

$$\begin{aligned} & \text{maximize} && \sum_{i=1}^n y_i \\ & \text{subject to} && (y_0, \dots, y_n) \in N_+^t(\text{cone}(IS(G))) \\ & && y_0 = 1 \end{aligned}$$

### 6.2.2 The Sherali–Adams Hierarchy

The Sherali–Adams hierarchy [55] defines a hierarchy of linear programs which give increasingly tighter relaxations. To see the intuition behind the hierarchy, we may view it as a strengthening of the LS procedure. Recall that the solution to a 0/1 integer program can be specified by a vector  $\mathbf{y} \in \{0, 1\}^n$ . In the Lovász–Schrijver hierarchy we defined auxiliary variables  $Y_{ij}$  and wanted to express the constraint that  $Y_{ij} = y_i \cdot y_j$ . We then expressed it by some implied linear conditions on the variables  $Y_{ij}$ .

Consider a solution  $(1, y_1, \dots, y_n)$  which is feasible at the second level of the LS hierarchy. Then the row  $Y_i$  of the protection matrix must also define a feasible solution to the “cone” version of the relaxation, say  $\text{cone}(IS(G))$ . Now, the solution  $\mathbf{y}' = Y_i = (Y_{i0}, Y_{i1}, \dots, Y_{in})$  must be feasible for the first level, and so there exists a protection matrix  $Y'$  for it. Now, we would also like to think of  $Y'_{jk}$  as a relaxation for  $y_i y_j y_k$ . However, notice that the choice of protection matrix  $Y'$  was dependent on the fact that we first chose the row  $Y_i$ . In particular, if we looked at the protection matrix  $Y''$  for the solution  $\mathbf{y}'' = Y_j = (Y_{j0}, Y_{j1}, \dots, Y_{jn})$ , it need not be true that  $Y'_{jk} = Y''_{ik}$ .

The Sherali–Adams hierarchy solves this problem by introducing all the auxiliary variables at once instead of by an inductive process. In particular, we define a variable  $Y_S$  for each  $S \subseteq [n]$  with  $|S| \leq t + 1$ . The intuition again is that we want to impose  $Y_S = \prod_{i \in S} y_i$ . However, we instead impose some linear conditions implied by this. For every constraint  $a^\top \mathbf{y} - b \leq 0$  of the starting LP relaxation,

maximize	$\sum_{i=1}^n Y_{\{i\}}$	
subject to	$\sum_{T' \subseteq T} (-1)^{ T' } \cdot [Y_{S \cup T' \cup \{j\}} + Y_{S \cup T' \cup \{i\}} - Y_{S \cup T'}] \leq 0 \quad  S  +  T  \leq t, (i, j) \in E$ $0 \leq \sum_{T' \subseteq T} (-1)^{ T' } \cdot Y_{S \cup T' \cup \{i\}} \leq \sum_{T' \subseteq T} (-1)^{ T' } \cdot Y_{S \cup T'} \quad  S  +  T  \leq t, i \in V$ $Y_\emptyset = 1$	

**Fig. 6.3** Sherali–Adams relaxation for Maximum Independent Set

we consider sets  $S, T$  such that  $|S| + |T| \leq t$  and impose a linear implication of  $(a^\top \mathbf{y} - b) \cdot \prod_{i \in S} y_i \cdot \prod_{j \in T} (1 - y_j) \leq 0$ , by requiring that

$$\sum_{T' \subseteq T} (-1)^{|T'|} \cdot \left( \sum_{i=1}^n a_i \cdot Y_{S \cup T' \cup \{i\}} - b \cdot Y_{S \cup T'} \right) \leq 0$$

Note again that the number of variables and constraints in the LP at level  $t$  is  $n^{O(t)}$  and hence it can be solved in time  $n^{O(t)}$ . Also, each such program is a relaxation, since for any  $\mathbf{y} \in \{0, 1\}^n$  satisfying the initial constraints,  $Y_S = \prod_{i \in S} y_i$  defines a valid level- $t$  solution. The program above gives the relaxation of Maximum Independent Set obtained at the  $t$ th level of the Sherali–Adams hierarchy (Fig. 6.3).

Since the above program is a convex relaxation, any convex combination of 0/1 solutions is also a solution to the program. It is convenient to think of the convex combination as defining a *distribution* over 0/1 solutions. With this interpretation, we can think of  $Y_S$  as the *probability* that all variables in set  $S$  are equal to 1. It is easy to show that the feasible sets for Sherali–Adams relaxations are characterized by solutions which “locally” (for every small subset of variables) look like valid distributions.

**Lemma 6.1.** *Consider a family of distributions  $\{\mathcal{D}(S)\}_{S \subseteq [n]: |S| \leq t+2}$ , where each  $\mathcal{D}(S)$  is defined over  $\{0, 1\}^S$ . If the distributions satisfy*

1. *For all  $(i, j) \in E$  and  $S \supseteq \{i, j\}$ ,  $\mathbb{P}_{\mathcal{D}(S)}[(y_i = 1) \wedge (y_j = 1)] = 0$ , and*
2. *For all  $S' \subseteq S \subseteq [n]$  with  $|S'| \leq t+1$ , the distributions  $\mathcal{D}(S')$ ,  $\mathcal{D}(S)$  agree on  $S'$ .*

*Then  $Y_S = \mathbb{P}_{\mathcal{D}(S)}[\bigwedge_{i \in S} (y_i = 1)]$  is a feasible solution for the above level- $t$  Sherali–Adams relaxation. Conversely, for any feasible solution  $\{Y'_S\}$  for the level- $(t+1)$  Sherali–Adams relaxation, there exists a family of distributions satisfying the above properties, as well as  $\mathbb{P}_{\mathcal{D}(S)}[\bigwedge_{i \in S} (y_i = 1)] = Y'_S$ , for all  $S' \subseteq S \subset [n]$  s.t.  $|S| \leq t+1$ .*

Extending further this intuition of the variables  $Y_S$  as probabilities, we can also define variables for arbitrary events over a set  $S$  of size at most  $t$ . A basic event is given by a *partial assignment*  $\alpha \in \{0, 1\}^S$  which assigns value 0 to some variables

maximize	$\sum_{i \in V} \ \mathbf{U}_{\{i\}}\ ^2$
subject to	$\langle \mathbf{U}_{\{i\}}, \mathbf{U}_{\{j\}} \rangle = 0 \quad \forall (i, j) \in E$ $\langle \mathbf{U}_{S_1}, \mathbf{U}_{S_2} \rangle = \langle \mathbf{U}_{S_3}, \mathbf{U}_{S_4} \rangle \quad \forall S_1 \cup S_2 = S_3 \cup S_4$ $\ \mathbf{U}_\emptyset\ ^2 = 1$

**Fig. 6.4** Lasserre SDP for Maximum Independent Set

in  $S$  (which we denote by  $\alpha^{-1}(0)$ ) and 1 to the others (denoted  $\alpha^{-1}(1)$ ). We can define variables  $X_{(S, \alpha)}$  when  $|S| \leq t$  and  $\alpha \in \{0, 1\}^S$  as

$$X_{(S, \alpha)} := \sum_{T \subseteq \alpha^{-1}(0)} (-1)^{|T|} Y_{\alpha^{-1}(1) \cup T}.$$

Note that the previous constraints imply that  $X_{(S, \alpha)} \geq 0$  for all  $(S, \alpha)$ . For a 0/1 solution, the intended values are  $X_{(S, \alpha)} = \prod_{i \in \alpha^{-1}(1)} y_i \prod_{i \in \alpha^{-1}(0)} (1 - y_i)$ . The previous program can easily be re-written in terms of the variables  $X_{(S, \alpha)}$ . This formulation in terms of variables for partial assignments also extends to problems where the variables in the integer program take values not in  $\{0, 1\}$  but in a larger finite domain  $[q]$ .

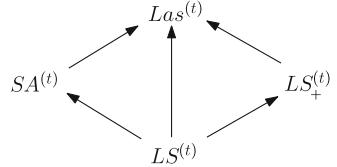
### 6.2.3 The Lasserre Hierarchy

The Lasserre hierarchy gives a sequence of increasingly tight semidefinite programming relaxations for a quadratic 0/1 program. As in the case of the Sherali–Adams hierarchy, the semidefinite program after  $t$  rounds of the Lasserre hierarchy also introduces a new (vector valued) variable for the product of every  $t$  variables in the original program.

For concreteness, we consider the program for Maximum Independent Set. The same procedure can be used to derive the level- $t$  SDP for any problem formulated as a quadratic integer program, with variables taking values in  $\{0, 1\}$ . Given a graph  $G = (V, E)$ , the integer program would have a variable  $X_i$  for each  $i \in V$  with  $y_i = 1$  if  $i$  is in the independent set and  $y_i = 0$  otherwise. To ensure that the solution is an independent set, we would enforce that  $y_i \cdot y_j = 0$  for all  $(i, j) \in E$ .

To obtain the Lasserre relaxation, we first think of an integer program which has a variable  $Y_S$  for each  $S \subseteq V, |S| \leq t$  where the intended solution, as before, is  $Y_S = 1$  iff all vertices in  $S$  are in the independent set. We can then add the constraint that the product  $Y_{S_1} \cdot Y_{S_2}$  must only depend on  $S_1 \cup S_2$ . For homogenization, we introduce an extra variable  $Y_\emptyset$  which is always supposed to be 1. Replacing the integer variables  $Y_S$  by vectors  $\mathbf{U}_S$  gives the semidefinite relaxation as below, where all sets  $S_i$  are assumed to be of cardinality at most  $t + 1$  (Fig. 6.4).

**Fig. 6.5** A comparison (the direction of the arrows denotes tighter relaxations)



Note that the program for level  $t$  only has vectors for sets of size at most  $t + 1$ . It can be shown that for any set  $S$  with  $|S| \leq t$ , the vectors  $\mathbf{U}_{S'}, S' \subseteq S$  induce a probability distribution over valid independent sets of the subgraph induced by  $S$ . However, unlike the Sherali–Adams hierarchy, the existence of such distributions is not a sufficient condition for the existence of a feasible solution for the semidefinite program.

As in the case of the Sherali–Adams hierarchy, one can also write the above program in terms of vectors  $\mathbf{V}_{(S,\alpha)}$  for partial assignments, which can be derived from the vectors  $\mathbf{U}_S$  (for 0/1 integer programs) as

$$\mathbf{V}_{(S,\alpha)} := \sum_{T \subseteq \alpha^{-1}(0)} (-1)^{|T|} \mathbf{U}_{\alpha^{-1}(1) \cup T}.$$

### 6.2.4 A Comparison

Let  $SA^{(t)}(\mathcal{P})$  denote the feasible set of the linear program obtained by starting from a basic linear relaxation  $\mathcal{P}$  (for some 0/1 program) and augmenting variables for  $t$  levels of the Sherali–Adams hierarchy. Similarly, let  $LS^{(t)}(\mathcal{P})$ ,  $LS_+^{(t)}(\mathcal{P})$ ,  $Las^{(t)}(\mathcal{P})$  represent feasible sets corresponding respectively to  $t$  levels of the LS, LS+ and Lasserre hierarchies. We summarize in the facts below, a comparison of these relaxations. The reader is referred to the excellent survey by Laurent [43] for a more detailed comparison.

1.  $LS^{(n)}(\mathcal{P}) = LS_+^{(n)}(\mathcal{P}) = SA^{(n)}(\mathcal{P}) = Las^{(n)}(\mathcal{P}) = \mathcal{I}$ , where  $\mathcal{I}$  denotes the convex hull of the 0/1 solutions to the starting integer program with  $n$  variables.
2. For all  $t \leq n$ ,  $LS^{(t)}(\mathcal{P}) \subseteq LS_+^{(t)}(\mathcal{P}) \subseteq Las^{(t)}(\mathcal{P})$ , and also  $LS^{(t)}(\mathcal{P}) \subseteq SA^{(t)}(\mathcal{P}) \subseteq Las^{(t)}(\mathcal{P})$ . Hence, the relaxations provided by the Lasserre hierarchy at each level are the strongest (most constrained) among the relaxations at the corresponding level of all the hierarchies discussed above (see Fig. 6.5).
3. If the starting relaxation  $P$  has  $n^{O(1)}$  constraints, then one can optimize over the sets  $LS^{(t)}(\mathcal{P})$ ,  $LS_+^{(t)}(\mathcal{P})$ ,  $SA^{(t)}(\mathcal{P})$  and  $Las^{(t)}(\mathcal{P})$  in time  $n^{O(t)}$ . This is known to be true for  $LS^{(t)}(\mathcal{P})$ ,  $LS_+^{(t)}(\mathcal{P})$  and  $SA^{(t)}(\mathcal{P})$  even if we only assume that  $\mathcal{P}$  has a weak separation oracle running in time  $n^{O(1)}$ . It is not known if one can optimize efficiently over  $Las^{(t)}(\mathcal{P})$  using an efficient separation oracle for  $\mathcal{P}$ .

### 6.3 Approximation Algorithms

The use of LP relaxations in approximation algorithms, as described in Sect. 6.1, is a well-established approach which has spawned much work and a large variety of techniques. On the other hand, the use of SDPs in approximation algorithms is a more recent development, starting with the seminal work of Goemans and Williamson on MAX-CUT [28]. This algorithm gave a better approximation than is achievable by known LP approaches (see Sect. 6.4), and placed MAX-CUT in what is now a large body of NP-hard problems for which a straightforward SDP-based algorithm gives an approximation which is conjectured to give the optimal approximation guarantee (see Sect. 6.5).

The main idea in the work of Goemans and Williamson [28] is to use the vectors arising from the Cholesky decomposition (defined in Sect. 6.2) of the PSD matrix of variables, and then to apply a randomized rounding technique now known as *hyperplane rounding*, which we will describe shortly. This approach (along with some refinements) quickly gave rise to improved approximation algorithms for a host of problems over the next decade.

A new approach, which included hyperplane rounding, but also introduced a subtler investigation of the geometry of finite metric spaces arising from feasible solutions to certain SDPs, was given in the celebrated work of Arora et al. [7] on **Sparsest Cut**. Among other things, this work showed that tightening SDPs by adding certain valid constraints (in this case, the triangle inequality for squared-distances) could yield improvements by significantly reducing the integrality gap of certain relaxations.

More generally, for any given NP-hard optimization problem, one may hope to gain improvements in the approximation guarantee by systematically strengthening an LP or SDP relaxation with additional constraints, as long as the strengthened relaxation can be solved in polynomial time. This sort of systematic strengthening is precisely what is offered by the various LP and SDP hierarchies described in Sect. 6.2, all of which produce polynomial time solvable convex relaxations in the first  $O(1)$  levels (in retrospect, such relaxations subsume certain important SDP relaxations, such as the **Sparsest Cut** relaxation in Sect. 6.3.2). Unfortunately, most results along these lines have been negative, showing that even relaxations at super-constant (and sometimes even linear) levels of certain hierarchies hardly yield any reduction in the integrality gap. These are discussed in Sect. 6.4.

In contrast, relatively few results, where improved approximation guarantees arise from the first  $O(1)$  levels of a hierarchy, have recently begun to emerge. Such positive results have been obtained, for example, for MAX-CUT in dense graphs [21], for Minimum Vertex Cover in planar graphs [48], and for MaxMin Allocation [8]. Later in this section, we will examine three such results. The first two, for Chromatic Number [18] and Hypergraph Independent Set [19] make explicit use of the interpretation of feasible solutions to relaxations arising from the Sherali–Adams (and by extension, Lasserre) hierarchy as families of distributions over local

0/1-assignments. Finally, we will consider the work of Karlin, Nguyen and Mathieu on Knapsack [34], which takes quite a different approach.

### 6.3.1 Max Cut and Hyperplane Rounding

The approximation algorithm of Goemans and Williamson [28] for MAX-CUT is perhaps the best known example of a simple SDP-based approximation algorithm which gives an approximation guarantee for the problem it approximates which is unmatched by any other method. Their algorithm gives an approximation ratio of  $1.138\dots$ , whereas until recently, all other known methods (including LP-based approaches – see Sect. 6.4) gave a 2-approximation in the worst case.<sup>3</sup> They propose the following SDP relaxation for the corresponding -1/1 program:

$$\begin{aligned} \text{maximize} \quad & \frac{1}{4} \sum_{(i,j) \in E} \|v_i - v_j\|^2 \\ \text{subject to} \quad & \|v_i\|^2 = 1 \quad \forall i \in V \end{aligned}$$

The rounding algorithm and analysis are equally simple and straightforward. Sample a vector  $z \in \mathbb{R}^n$  uniformly at random from the unit sphere  $\{u \in \mathbb{R}^n \mid \|u\| = 1\}$ , and output cut  $(S, \bar{S})$ , where  $S = \{i \in V \mid \langle z, v_i \rangle > 0\}$ . That is, separate the vectors on either side of the hyperplane orthogonal to  $z$ , and take the corresponding cut in  $G$ .

It is not hard to see that the probability that any two vectors are separated is proportional to the angle between them (where for the maximal angle,  $\pi$ , the vectors are separated with probability 1). For any edge  $(i, j) \in E$ , let  $\theta_{ij} = \frac{1}{4}\|v_i - v_j\|^2$  be its contribution to the objective function. Then the probability that this edge will be cut is at least

$$\frac{\arccos\langle v_i, v_j \rangle}{\pi} = \frac{\arccos(1 - 2\theta_{ij})}{\pi} \geq \theta_{ij} \cdot \min_{0 < \theta \leq 1} \frac{\arccos(1 - 2\theta)}{\theta} = C_{\text{GW}} \cdot \theta_{ij},$$

where  $C_{\text{GW}} = 0.878\dots$  is the Goemans–Williamson constant. It follows immediately, by linearity of expectation, that the expected size of the cut is at least a factor  $C_{\text{GW}}$  times the value of the objective function of the SDP, thus giving a  $1/C_{\text{GW}} \approx 1.138$  approximation.

Surprisingly, there is now some evidence that the Goemans–Williamson constant  $C_{\text{GW}}$  is not simply an artifact of the above analysis, but in fact the best possible. That is, assuming a conjecture about the hardness of a certain problem known as **Unique Games**, achieving an approximation of  $1/C_{\text{GW}} - \varepsilon$  is computationally intractable for any  $\varepsilon > 0$ . The Unique Games problem has similar consequences for many optimization problems, discussed further in Sect. 6.5.

---

<sup>3</sup>More recently,  $1.6281\dots$ -approximations were obtained using spectral techniques [56, 58] and combinatorially using random walks [32].

$$\text{minimize} \sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \quad (6.1)$$

$$\text{subject to} \quad \|v_i - v_j\|^2 + \|v_j - v_k\|^2 \geq \|v_i - v_k\|^2 \quad \forall i, j, k \quad (6.2)$$

$$\sum_{i < j} \|v_i - v_j\|^2 = 1 \quad (6.3)$$

**Fig. 6.6** Sparsest Cut SDP relaxation

### 6.3.2 Sparsest Cut and Metric Embeddings

A common feature of the MAX-CUT algorithm above, and other SDP-based algorithms that followed, is the use of “local” SDP relaxations. In a local relaxation, every SDP constraint involves a single combinatorial item – a vertex, an edge, or a clause (in a Constraint Satisfaction Problem). In turn, the analysis of the rounding algorithm involves a local examination of the (expected) contribution of each item to the rounded solution, either taking into account the local constraints for that item, or comparing its contribution in the rounded solution to its contribution in the objective function of the SDP (as in the MAX-CUT algorithm). The bound on the approximation ratio then follows directly from linearity of expectation.

We find a departure from this approach in the work of Arora et al. [7] on Sparsest Cut. This problem is defined as follows: Given a graph  $G = (V, E)$ , find a cut  $(S, \bar{S})$  that minimizes the ratio

$$\frac{|E(S, \bar{S})|}{|S||\bar{S}|}.$$

Their algorithm gives a  $\sqrt{\log n}$ -approximation, which relies on the above SDP relaxation (Fig. 6.6):

This relaxation is derived from the integer programming formulation as follows: in a -1/1 solution, the objective function (6.1) and the left hand side of (6.3) would represent  $4|E(S, \bar{S})|$  and  $4|S||\bar{S}|$ , respectively. We then scale the solution so that (6.1) represents the ratio of the two expressions above. Note the use of the triangle inequality for squared-distances (6.2), which is already a slightly non-local constraint, as it involves more than simply the variables associated with a single vertex or edge. In fact, the triangle inequality is necessary, since otherwise the integrality gap may be as large as  $\Omega(n)$ . To see this, consider the case of the undirected cycle  $C_n$  with vertices  $\{0, \dots, n-1\}$  and edges  $\{(i, j) \mid j = i + 1 \pmod n\}$ . It is easy to see that the sparsest cut has sparsity  $8/(n-2)^2$ . On the other hand, without triangle inequality, the above relaxation has the following 2-dimensional solution:  $v_j = \frac{1}{n}(\cos(2j\pi/n), \sin(2j\pi/n))$ , for which the objective function has value  $(4 - o(1))\pi^2/n^3$ .

1. Let  $z = (z_1, \dots, z_n)$  where the coordinates are independent Gaussians  $z_i \sim \mathcal{N}(0, 1)$ .
2. For some constant  $c > 0$ , let  $S' = \{i \mid \langle z, v_i \rangle \geq c/2\}$  and  $T' = \{j \mid \langle z, v_j \rangle \leq -c/2\}$ .
3. While there is any pair  $(i, j) \in S' \times T'$  such that  $\|v_i - v_j\|^2 < c^2\delta$ , remove  $i$  and  $j$  from  $S', T'$ .
4. Output the remaining sets  $S = S'$  and  $T = T'$ .

**Fig. 6.7** Algorithm ARV-Round( $\delta$ )

The triangle-inequality constraint (6.2) means that the vectors  $\{v_i\}$  form a finite metric space, known as  $L_2^2$ , or *negative type* metric. A crucial component of the algorithm and analysis is the following structure theorem for certain  $L_2^2$  metrics (slightly simplified here):

**Theorem 6.1.** *Let  $\{v_i \mid i \in [n]\}$  be a set of vectors in  $\mathbb{R}^n$  of constant length (i.e. which all belong to some annulus  $\{v \in \mathbb{R}^n \mid 0 < c_1 < \|v\| < c_2\}$ ) which form an  $L_2^2$  metric, and which satisfy*

$$\sum_{i < j} \|v_i - v_j\|^2 \geq n^2/3. \quad (6.4)$$

*Then there is a polynomial-time randomized algorithm which outputs two disjoint subsets  $S, T \subset [n]$  both of size  $\Omega(n)$ , such that for every  $i \in S$  and  $j \in T$  we have*

$$\|v_i - v_j\|^2 \geq 1/\sqrt{\log n}. \quad (6.5)$$

Consider the algorithm in Fig. 6.7, which takes as a parameter  $\delta > 0$ , the intended  $L_2^2$  distance between  $S$  and  $T$ .

Note that the distribution of vector  $z$  is rotationally invariant. Thus for any fixed unit vector  $v \in \mathbb{R}^n$ , we have  $\langle z, v \rangle \sim \mathcal{N}(0, 1)$ . Using this fact, the constant density of the standard normal distribution near 0, assumption (6.4), and Markov's inequality, an easy argument shows that for an appropriate constant  $c > 0$ , Step 2 yields sets  $S'$  and  $T'$  of size  $\Omega(n)$  with constant probability. Thus, it suffices to show that in Step 3 only  $o(n)$  pairs are removed in expectation for  $\delta = \Theta(1/\sqrt{\log n})$ .

Let us first see that a “local” analysis, based on linearity of expectation, works for  $\delta = O(1/\log n)$ . By the properties of Gaussian distribution, we have the following tail bound on the projection of any fixed vector  $v \in \mathbb{R}^n$ , for  $C \geq \|v\|$ :

$$\mathbb{P}[\langle z, v \rangle \geq C] \leq e^{-\frac{1}{2}C^2/\|v\|^2} \quad (6.6)$$

Note that every pair  $(i, j) \in S' \times T'$  satisfies  $\langle z, v_i - v_j \rangle \geq c$ . By (6.6) the probability that this occurs for any pair that is sufficiently close to be eliminated in Step 3 is at most  $e^{-1/(2\delta)}$ , which is at most  $1/n^3$  for some  $\delta = O(1/\log n)$ . Thus, by linearity of expectation, the expected number of pairs eliminated in this case is  $o(1)$ .

To show that the algorithm works even for  $\delta = \Theta(1/\sqrt{\log n})$  requires a much more subtle argument. We give an overview of the simplified proof in [45], with some technical details omitted. We will need the following notation. For  $i \in [n]$  let us denote  $\Gamma(i) = \{j \mid \|v_j - v_i\|^2 < c^2\delta\}$ , for a set  $I \subseteq n$  denote  $\Gamma(I) = \bigcup_{i \in I} \Gamma(i)$ , and let

$\Gamma^t(i)$  be  $\Gamma(\dots(\Gamma(i))\dots)$ , applied  $t$  times. Finally, denote by  $R^t(i)$  the event “ $\exists j \in \Gamma^t(i)$  s.t.  $\langle z, v_j - v_i \rangle \geq ct$ ”. Note that  $i$  is eliminated in Step 3 only when the event  $R^1(i)$  occurs. At a high level, the analysis relies on the following idea introduced in [7], known as a *chaining argument*: By way of contradiction, assume that every  $i \in [n]$  is eliminated at step 3 with probability  $\mathcal{Q}(1)$  (this condition can be shown to be essentially equivalent to algorithm ARV-Round failing). That is, for every  $i \in [n]$ , event  $R^1(i)$  occurs with constant probability. Suppose we have already shown that  $R^{t-1}(j)$  occurs with constant probability for all  $j$ . Then for any  $i \in [n]$ , with constant probability there is some  $j \in \Gamma(i)$  because of which  $R^1(i)$  occurs, and with constant probability  $R^{t-1}(j)$  occurs for this particular  $j$ . Together, for some  $k \in \Gamma^{t-1}(j) \subseteq \Gamma^t(i)$  the two events together imply

$$\langle z, v_k - v_i \rangle = \langle z, v_k - v_j \rangle + \langle z, v_j - v_i \rangle \geq c(t-1) + c = ct,$$

or in other words, the event  $R^t(i)$ .

To summarize, the chaining argument, applied for  $t$  steps, says that if algorithm ARV-Round fails, then for every vertex  $i \in [n]$ , event  $R^t(i)$  occurs with constant probability. This has the following immediate implication:

**Lemma 6.2.** *If the chaining argument works after  $t$  steps, then ARV-Round works for  $\delta = \mathcal{Q}(t/\log n)$ .*

*Proof.* By way of contradiction, assume that ARV-Round fails for this  $\delta$ . Then applying the chaining argument for  $t$  steps, we get, for all  $i \in [n]$ ,

$$\mathbb{P}[\exists j \in \Gamma^t(i) : \langle z, v_j - v_i \rangle \geq ct] = \mathcal{Q}(1). \quad (6.7)$$

We now make crucial use of the  $L_2^2$  property (6.2): it implies that for all  $j \in \Gamma^t$  we have  $\|v_j - v_i\|^2 < c^2\delta t$ . Thus, bound (6.6) and a union bound imply

$$\mathbb{P}[\exists j \in \Gamma^t(i) : \langle z, v_j - v_i \rangle \geq ct] \leq ne^{-t/(2\delta)},$$

which contradicts (6.7).  $\square$

For how many steps can we continue the above chaining argument? The main obstacle to making the argument rigorous is that, even though the events  $R^t(j)$  for the various  $j \in \Gamma(i)$  have constant probability, we cannot chain them to the event  $R^1(i)$  since there is no guarantee that any of them will intersect the event  $R^1(i)$ . Instead, we have to settle for slightly weaker events  $\tilde{R}^t(j)$  which have probability close to 1. Let  $\tilde{R}^t(j)$  be the event “ $\exists k \in \Gamma^t(j)$  s.t.  $\langle z, v_k - v_j \rangle \geq c(t+1)/2$ ”. By considerations of measure concentration, it can be shown that if  $\tilde{R}^t(j)$  occurs with constant probability, then

$$\mathbb{P}[\exists k \in \Gamma^t(j) : \langle z, v_k - v_j \rangle \geq ct/2] \geq 1 - e^{-1/(9t\delta)},$$

as long as  $t\delta \leq c'$  for some constant  $c' > 0$ . If these events have probability close to 1, we can then chain them with the event  $R^1(i)$  to imply the event  $R^{t+1}(i)$ . For the above probability to be close to 1, it suffices to require that  $t\delta \leq c''$  for some

constant  $c'' > 0$ . Thus the chaining argument works for  $t = \mathcal{O}(1/\delta)$  steps, which by Lemma 6.2 means that ARV-Round works for  $\delta = \mathcal{O}(1/\sqrt{\log n})$ .

The Sparsest Cut problem has a generalization known as the *general demand* version of Sparsest Cut. In this variant, every pair of nodes  $i, j \in V$  has some edge weight  $w_{ij} \geq 0$  ( $w_{ij} = 0$  for non-edges) and some demand  $d_{ij} \geq 0$ , and the objective is to find a cut  $(S, \bar{S})$  minimizing the ratio  $\sum_{i \in S, j \in \bar{S}} w_{ij} / \sum_{i \in S, j \in \bar{S}} d_{ij}$ . For any graph  $G$ , the worst case integrality gap of the standard LP relaxation for this problem on  $G$  (over all choices of edge weights and demands) is known to be equivalent to the  $L_1$ -embeddability of that graph (the least distortion with which a shortest-path metric on  $G$  can be embedded into  $L_1$ , maximized over all possible edge lengths). One might also ask whether the SDP relaxation also has connections to embeddings into  $L_1$ . Indeed, in later work, Arora et al. [3] combined Theorem 6.1 with a careful accounting scheme, to show that any  $n$ -point  $L_2^2$  metric embeds into  $L_2$  (and hence into  $L_1$ ) with distortion  $O(\sqrt{\log n} \log \log n)$ .

### 6.3.3 Chromatic Number: Coloring 3-Colorable Graphs

The 3-Coloring problem is a classic NP-complete problem. Its optimization variant, Chromatic Number is NP-hard to approximate within an  $n^{1-\varepsilon}$ -factor for any constant  $\varepsilon > 0$  [23, 60]. Therefore, there has been much focus on approximation algorithms when the chromatic number is small. Specifically, we are interested in the following question: given a 3-colorable graph (where the coloring itself is not given), what is the least number of colors with which we can efficiently legally color the graph?

All algorithms for this problem (or subroutines thereof) involve some assumption on either the minimum or maximum degree in the graph. For simplicity of presentation, let us only consider  $d$ -regular graphs here. After a series of purely combinatorial algorithms, Karger et al. [33] gave an  $\tilde{O}(d^{1/3})$ -coloring for such graphs ( $\tilde{O}(\cdot)$  hides polylogarithmic factors) in one of the early SDP-based algorithms following [28]. Subsequently and Blum and Karger [11] combined this with an earlier combinatorial  $\tilde{O}((n/d)^{3/5})$ -coloring algorithm of Blum [12] to give a  $\tilde{O}(n^{3/14})$ -coloring (or roughly  $n^{0.2143}$ ).

This was the state of the art until a series of two papers, by Arora et al. [6], and by Chlamtac [18] improved the above guarantee to roughly  $n^{0.2111}$  and  $n^{0.2072}$ , respectively, by carefully characterizing the tight case of the analysis of the SDP rounding in [33] and showing it cannot occur (at least near the Blum–Karger threshold of  $d = n^{9/14}$ ). Whereas Arora et al. [6] achieve this by giving a chaining and measure-concentration based argument reminiscent of [7], Chlamtac [18] makes explicit use of a level-2 Lasserre relaxation and its implicit interpretation as a family of local-distributions (as discussed earlier). Let us first examine the various SDP relaxations used in these works, before giving a brief high-level description of the intuition behind the Lasserre hierarchy-based improvement in [18].

Karger et al. [33] proposed the following SDP relaxation for Chromatic Number of a graph  $G = (V, E)$ , which they called the *vector chromatic number*:

$$\text{minimize } \kappa \tag{6.8}$$

$$\text{subject to } \|v_i\|^2 = 1 \quad \forall i \in V \tag{6.9}$$

$$\langle v_i, v_j \rangle \leq -\frac{1}{\kappa-1} \quad \forall (i, j) \in E \tag{6.10}$$

This relaxation is based on the following observation: For any  $k \in \mathbb{N}$ , there is a set of  $k$  unit-vectors whose pairwise inner-products are all  $-1/(k-1)$ , and moreover this is the minimum value for which such vectors exist. The vector chromatic number is equivalent to  $\vartheta'(\overline{G})$ , where  $\overline{G}$  is the complement graph, and  $\vartheta'$  is the variant of the Lovász  $\vartheta$ -function introduced by Schrijver [54].<sup>4</sup>

Let us focus on relaxations for 3-Coloring. Since the above relaxation is not based on a 0/1 program, we require some manipulation in order to strengthen it using an SDP hierarchy based on 0/1 programs, such as the Lasserre hierarchy. Such an approach has been suggested by Gvozdenović and Laurent [29], based on the following observation of Chvátal [20]: Let  $X$  be a set of three colors  $X = \{R, B, Y\}$ . For a graph  $G = (V, E)$ , define a new graph  $G_X = (V_X, E_X)$ , where  $V_X = V \times X$ , and  $E_X = \{((i, C), (j, C)) \mid (i, j) \in E \text{ and } C \in X\} \cup \{((i, C_1), (i, C_2)) \mid i \in V \text{ and } C_1 \neq C_2 \in X\}$ . Then  $G$  is 3-colorable iff  $G_X$  contains an independent set of size  $n$  (note that it can never contain a larger independent set). Moreover, every independent set  $I$  corresponds to a unique 3-coloring  $f_I : V \rightarrow X$  in the natural way:  $f_I(i) = C$  for all  $(i, C) \in I$ . We can now apply any of the SDP hierarchies described earlier for Maximum Independent Set to the graph  $G_X$ .

How does such a hierarchy give improvements over the algorithm and analysis in [11, 33]? It suffices to consider the tight (or nearly tight) case of the SDP rounding in [33], and show that when the analysis is tight, there is an alternative rounding which performs vastly better than all currently known algorithms. To understand the tight case of the analysis, it helps to examine the interpretation of the SDP solution as a family of distributions on local colorings (as arise in Sherali–Adams and Lasserre relaxations, e.g. as in Lemma 6.1). In particular, let us fix a single vertex  $i$ , and some coloring, say  $(i, R)$ , and consider the random color assignments (over  $\{B, Y\}$ ) to its various neighbors  $N(i)$ . A simple probabilistic argument says that it cannot be the case that most pairs of vertices  $j, k \in N(i)$  are assigned the same color with probability much less than  $\frac{1}{2}$  (since the average correlation cannot be very negative). On the other hand, it turns out that in the tight case, these probabilities cannot be much larger on average. Indeed, if many such pairs are assigned the same color with significantly higher probability, then the corresponding vectors  $u_j, u_k$  will have a larger inner product. At a very high level, this facilitates a better

---

<sup>4</sup>The Lovász  $\vartheta$  function itself is actually equivalent to a variant of the above relaxation, called the *strict vector chromatic number*, in which we have equality in (6.10).

SDP rounding, since when the SDP solution is more clustered, it becomes easier to separate vertices into fewer color classes. To summarize, if the SDP analysis of [33] is tight, then in the interpretation of the SDP solution locally as a distribution on colorings, most vertices at distance two will be assigned the same color with probability  $\approx \frac{1}{2}$ .

When can a distribution on 3-colorings with this property exist? Let us fix two vertices  $j, k$  at distance 2 from each other which are assigned the same color with probability  $\frac{1}{2}$ , which also have many common neighbors  $i \in N(j) \cap N(k)$ . We know that when  $j$  and  $k$  are assigned different colors, say  $B$  and  $Y$ , then all vertices in  $N(j) \cap N(k)$  will have *the same* color as each other (here  $R$ ), while if  $j$  and  $k$  are assigned the same color, then as before, most pairs of vertices  $i, i' \in N(j) \cap N(k)$  will have the same color with probability  $\approx \frac{1}{2}$ . Thus, most pairs of common neighbors  $i, i'$  will receive the same color with probability  $> \frac{1}{2}$ , contradicting our earlier assertion for vertices at distance 2. The only resolution of this contradiction is for all common neighborhoods  $N(j) \cap N(k)$  to all be quite small. A simple counting argument then gives a lower bound on the size of any 2-neighborhood  $N(N(j))$ .

The precise formalization of the above argument gives a bound of  $|N(N(j))| \geq d^{3/2}$ . This is already a contradiction when  $d > n^{2/3}$  (thus ruling out any tight case of the previous analysis for this range of parameters). However, when the degree  $d$  is near the Blum–Karger threshold of  $n^{9/14}$ , this is not quite enough. In this case we make use of the fact that each vertex in  $N(N(j))$  has the same color as  $j$  with probability  $\frac{1}{2}$ , which implies an independent set of size  $\frac{1}{2}|N(N(j))|$  in the 2-neighborhood. Standard Minimum Vertex Cover approximations then allow us to extract a large independent set (a color class in our coloring), here of size  $\tilde{\mathcal{O}}(d^{3/2}) = \tilde{\mathcal{O}}(n^{27/28})$ , which makes progress towards an  $\tilde{\mathcal{O}}(n^{1/28})$  coloring. To summarize, this argument shows that either the analysis of the SDP rounding in [33] is not tight (and then the algorithm performs better), or a different algorithm yields a far better coloring than the current approach. Formalizing the argument involves a careful analysis of the SDP rounding in [33], and is beyond the scope of this chapter.

### 6.3.4 Mixed Hierarchies and Hypergraph Independent Set

In all SDP-based approximation algorithms, such as the MAX-CUT and Sparsest Cut algorithms we've seen, the rounding algorithm and analysis rely on the geometry of SDP solutions. More recently, algorithms such as the coloring algorithm of [18] also use the interpretation of solutions to Lasserre hierarchies as families of distributions. However, this interpretation also arises in LP hierarchies such as Sherali–Adams. We might ask whether weaker SDP hierarchies than Lasserre, which combine an LP characterized by local distributions with a simple SDP could also yield improved approximations. Such mixed hierarchies arise naturally in approximation algorithms. In fact, under certain complexity theoretic assumptions, they already give optimal approximations for  $k$ -CSPs at level  $k$  (see Sect. 6.5).

$$\begin{aligned}
& \text{maximize} && \sum_{i=1}^n Y_{\{i\}} \\
& \text{subject to} && Y_{\{i,j,k\}} = 0 \quad (i, j, k) \in E \\
& && (Y_S)_{S \subseteq T} \in I_t \quad |T| = t \\
& && M_1(Y) \geq 0
\end{aligned}$$

**Fig. 6.8** Mixed level- $t$  relaxation for Hypergraph Independent Set

One example where such a hierarchy gives an infinite sequence of improved approximation guarantees is described in the work of Chlamtac and Singh [19] on Hypergraph Independent Set. Let us focus on the 3-uniform variant of this problem: We are given a hypergraph  $H = (V, E)$ , where the hyperedges in  $E$  are all subsets  $e \subset V$  of cardinality  $|e| = 3$ . Find the maximum independent set in  $H$ , where an independent set is any subset  $S \subseteq V$  that does not contain any hyperedge as a subset.

Let us now define a mixed hierarchy of relaxations for this problem. First, for any integer  $t \geq 1$ , define

$$I_t = \text{conv}\left(\left\{(\prod_{i \in S} x_i)_{S \subseteq [t]} \mid x \in \{0, 1\}^t\right\}\right).$$

Note that there is a one-to-one correspondence between vectors in  $I_t$  and distributions over assignments  $f : [t] \rightarrow \{0, 1\}$ . Now, define  $M_1(Y)$  for any vector  $(Y_S)$  to be the  $(n+1) \times (n+1)$  moment matrix  $(Y_{S \cup T})_{|S|, |T| \leq 1}$ . For  $t \geq 3$ , Fig. 6.8 (above) gives a level- $t$  relaxation for Hypergraph Independent Set. Note that the constraint  $M_1(Y) \geq 0$  simply means that there exists a set of  $n+1$  vectors  $v_0, v_{\{1\}}, \dots, v_{\{n\}}$  whose pairwise inner-products are consistent with the LP values, i.e.  $\langle v_S, v_T \rangle = Y_{S \cup T}$ .

Like Chromatic Number, the problems Maximum Independent Set and Hypergraph Independent Set are notoriously hard to approximate. Therefore, for polynomial-size relaxations, we cannot expect integrality gaps smaller than  $n^{1-o(1)}$  (unless P = NP). Instead, we focus on integrality gaps parametrized by the optimum value of the relaxation. As before, denote by **FRAC** and **OPT** the optimum of the LP or SDP relaxation, and the 0/1 optimum, respectively. Then for  $\gamma \in [0, 1]$  we can define the approximation curve of the relaxation to be

$$\text{MINOPT}_n(\gamma) = \min\{\text{OPT} \mid \text{FRAC} \geq \gamma n\},$$

where the minimization is over all problem instances on  $n$  variables. Thus, the approximation curve can be meaningful even when the integrality gap (in this case,  $\max_\gamma (\gamma n / \text{MINOPT}(\gamma))$ ) is  $\tilde{\Omega}(n)$ .

For example, a series of papers [2, 30] shows that for Maximum Independent Set, the SDP relaxation given by the Lovász  $\vartheta$  function has an approximation curve of  $\text{MINOPT}_n(\gamma) = \tilde{\mathcal{Q}}(n^{f(\gamma)})$ , where  $f$  satisfies  $f(\gamma) \geq 3\gamma/(1+\gamma)$  for  $1/\gamma \in \mathbb{N} \setminus \{1\}$ ,

and  $f(\gamma) \geq \min\{2\gamma, 1\}$  in general. Attempts to extend this approach to Hypergraph Independent Set by Krivelevich et al. [41] yielded only partial results. In particular, for all  $\gamma \leq \frac{1}{2}$  the approximation curve given by their SDP relaxation for this problem is quite poor:  $\text{MINOPT}_n(\gamma) = 2$ . This shortcoming is in fact inherent. The SDP relaxation arising from level-3 of the mixed hierarchy in Fig. 6.8, which is at least as strong as the SDP used by [41], always has optimum value at least  $n/2$ .

While no fixed level of the mixed hierarchy gives non-trivial guarantees for all  $\gamma > 0$ , the range of  $\gamma$  for which the approximation curve is non-trivial grows as we use higher levels, and tends to  $[0, 1]$  in the limit. Let us denote by  $\text{MINOPT}_{n,t}$  the value of  $\text{MINOPT}_n$  as defined above with respect to the level- $t$  mixed hierarchy in Fig. 6.8. Then the results of [19] for this hierarchy can be summarized as follows:

$$A_{n,t}(\gamma) \leq \text{MINOPT}_{n,t}(\gamma) \leq B_{n,t}(\gamma),$$

where

$$A_{n,t}(\gamma) = \begin{cases} 2, & \gamma < \frac{2}{\sqrt{t}} \\ n^{\gamma^2/32}, & \gamma \geq \frac{2}{\sqrt{t}} \end{cases} \quad B_{n,t}(\gamma) = \begin{cases} 2, & \gamma \leq \frac{1}{t-1} \\ n, & \gamma > \frac{1}{t-1} \end{cases}.$$

Note that the second case in bound  $A_{n,t}(\gamma)$  together with the first case in bound  $B_{n,t}(\gamma)$  shows that indeed, there is an infinite sequence of strict improvements in the approximation guarantee as  $t$  increases (for instance,  $\text{MINOPT}_{n,9t^2}(\gamma)$  is much larger than  $\text{MINOPT}_{n,t+1}(\gamma)$  in the range  $2/(3t) \leq \gamma \leq 1/t$ ). While slightly better parameters for  $A_{n,t}(\gamma)$  are known for the Lasserre hierarchy, there are currently no bounds such as  $B_{n,t}(\gamma)$  for this hierarchy (in particular, no bounds which would preclude the possibility that even the level-2 Lasserre relaxation gives a non-trivial guarantee for every  $\gamma > 0$ ).

### 6.3.5 Sherali–Adams and Lasserre Relaxations for Knapsack

The Knapsack problem is defined as follows: Given a set of  $n$  items with costs  $c_i \geq 0$  and rewards  $r_i \geq 0$ , and some capacity  $C > 0$ , find a subset  $S \subseteq [n]$  of cost  $\sum_{i \in S} c_i$  at most  $C$  which maximizes the reward  $\sum_{i \in S} r_i$ . This is a well-understood classic NP-complete problem which is easy to approximate – it admits an FPTAS [31, 44]. While from the perspective of approximation algorithms, there is nothing to be gained by applying convex optimization techniques to this problem, it is a useful tool for gaining a better understanding of the strengths and properties of various hierarchies of relaxations. Let us review the results of Karlin et al. [34], who recently investigated this topic.

Consider the natural LP relaxation for Knapsack (Fig. 6.9):

$$\text{maximize} \sum_i r_i y_i \quad (6.11)$$

$$\text{subject to} \quad \sum_i c_i y_i \leq C \quad (6.12)$$

$$0 \leq y_i \leq 1 \quad \forall i \in [n] \quad (6.13)$$

**Fig. 6.9** Standard LP relaxation for Knapsack

This LP relaxation has an integrality gap of 2. Due to the existence of simple combinatorial  $(1+\varepsilon)$ -approximations for Knapsack, one would expect that strengthening this relaxation would quickly reduce the integrality gap. Nevertheless, the authors of [34] show that this is not the case for Sherali–Adams:

**Theorem 6.2.** *For any  $0 < \delta < \frac{1}{3}$ , the integrality gap of the level- $\delta n$  Sherali–Adams relaxation for Knapsack is at least  $2/(1+\delta)$ .*

This follows by considering a simple instance of Knapsack: let all the costs and rewards be 1, and the capacity be  $C = 2 - \varepsilon$  for some sufficiently small  $\varepsilon > 0$ . The optimum in this instance has reward 1, while the LP admits a solution of value  $(2 - \varepsilon)/(1 + (1 - \varepsilon)\delta)$  (the lemma follows by letting  $\varepsilon$  tend to 0). To see this, consider the Sherali–Adams solution  $Y_\emptyset = 1$ ,  $Y_{\{i\}} = p$  for all  $i \in [n]$ , where  $p = (2 - \varepsilon)/((1 + (1 - \varepsilon)\delta)n)$ , and  $Y_I = 0$  for all  $|I| > 1$ . This clearly gives valid distributions over assignments to sets of size  $1/p > \delta n$ . It can be checked that for this solution, the only relevant constraints that need to be verified are certain lifts of constraint (6.12). Specifically, under the interpretation of Sherali–Adams as a relaxation for a global distribution over knapsack solutions  $K \subseteq [n]$ , for all  $J \subseteq [n]$  of cardinality  $|J| \leq \delta n$ , we need to verify the constraint corresponding to  $\mathbb{E}[|K| \mid J \cap K = \emptyset] \leq C$ . But for the above LP solution, the expression corresponding to this expectation is simply  $(n - |J|)p/(1 - |J|p) \leq 2 - \varepsilon$  (assuming  $|J| \leq \delta n$ ).

Why is the integrality gap so large after so many rounds? The main reason is that Sherali–Adams is very inconsistent with respect to the value of the objective function. Specifically, in the above example, the objective function takes value  $\sum_i Y_{\{i\}}$  which is nearly 2. However, if we lift this expression by one round by “conditioning on  $x_j = 1$ ” for a fixed  $j \in [n]$ , we get  $(Y_{\{j\}} + \sum_{i \in [n], i \neq j} Y_{\{i,j\}})/Y_{\{j\}} = 1$ . We can circumvent this problem by rephrasing the initial LP as a feasibility LP, with the objective function as an added constraint, as in Fig. 6.10. Note that  $R$  in constraint (6.14) is an external parameter, and not an LP variable. We can now take this relaxation and apply Sherali–Adams to it, and find the maximum  $R$  that for which the new Sherali–Adams relaxation is feasible (say, by binary search).

Applying Sherali–Adams to the feasibility LP above significantly reduces the integrality gap:

**Theorem 6.3.** *The integrality gap of level- $t$  of the Sherali–Adams hierarchy applied to the feasibility LP in Fig. 6.10 is at most  $1 + 1/(t-2)$ .*

	find $y_1, \dots, y_n$	
which satisfy	$\sum_i r_i y_i \geq R$	(6.14)
	$\sum_i c_i y_i \leq C$	(6.15)
	$0 \leq y_i \leq 1$	$\forall i \in [n]$ (6.16)

**Fig. 6.10** Parametrized feasibility LP relaxation for Knapsack

To see why lifting the objective function helps, we first need the following well-known lemma (here, Greedy is the value of the combinatorial greedy algorithm for Knapsack, which does not use any convex relaxations of the problem):

**Lemma 6.3.** *The standard LP relaxation satisfies  $\text{FRAC} \leq \text{Greedy} + \max_i r_i (\leq 2\text{OPT})$ .*

The bound in Theorem 6.3 now follows from a simple rounding algorithm: Let  $S_{t-1} = \{i \mid r_i > \text{OPT}/(t-1)\}$  (we can guess this set in  $n$  trials by sorting). At most  $(t-2)$  items in  $S_{t-1}$  can fit in the knapsack (otherwise OPT would be higher). So, as long as there exists an item  $i \in S_{t-1}$  with non-zero LP value, condition the LP solution on picking this item. After at most  $(t-1)$  steps, all the remaining items in  $S_{t-1}$  have LP value 0. Let  $K_0$  be the set of items picked so far, with reward  $R_0 = r(K_0)$ , and consider the current LP solution  $\{Y'\}$  restricted to  $[n] \setminus S_{t-1}$ . Since we lifted the objective function, the value of this LP is now  $\sum_{i \notin S_{t-1}} Y'_i \geq R - R_0$ . Now apply the greedy algorithm to the remaining items, giving some additional reward  $R_g$ . By the above bound on the LP value, and Lemma 6.3, we have

$$R - R_0 \leq \sum_{i \notin S_{t-1}} Y'_i \leq R_g + \max_{i \notin S_{t-1}} r_i < R_g + \text{OPT}/(t-1) \leq R_g + R/(t-1).$$

Therefore, the value of the rounded solution is  $R_0 + R_g \geq R - R/(t-1) = R(1 - 1/(t-1))$ .

Surprisingly, the Lasserre hierarchy does not require such manipulations. Applying it directly to the standard (maximization) LP immediately yields essentially the same guarantee as in Theorem 6.3:

**Theorem 6.4.** *Level- $t$  of the Lasserre hierarchy applied to the standard relaxation for Knapsack has integrality gap  $\leq 1 + \frac{1}{t-1}$ .*

This follows from a similar rounding algorithm as above. We need to adapt the rounding of the fractional solution for the high-reward items  $S_t$  to the current setting, after which we can apply the greedy algorithm to the remaining items, just as before. Note that for the same analysis to go through, the rounding for  $S_t$  should be lossless, in the sense that it does not decrease the overall objective value.

The existence of a lossless rounding for  $S_t$  follows immediately from the following key observation: the solution to the above SDP restricted to the set  $S_t$

is already in the *integral hull*. To see this, consider any set  $S$  for which the **SDP** solution satisfies  $\|\mathbf{U}_T\| = 0$  for all subsets  $T \subseteq S$  of cardinality  $|T| = t$ . Then we can add zero vectors  $\mathbf{U}_T = 0$  for *all* subsets  $T \subseteq S$  of cardinality  $|T| > t$ , without violating the essential consistency constraints

$$\forall T_1, T_2, T_3, T_4 \subseteq S \text{ s.t. } T_1 \cup T_2 = T_3 \cup T_4 : \langle \mathbf{U}_{T_1}, \mathbf{U}_{T_2} \rangle = \langle \mathbf{U}_{T_3}, \mathbf{U}_{T_4} \rangle.$$

This would extend the current solution to a valid level- $|S|$  Lasserre solution. Such a solution must be in the integral hull (as would be any level- $|S|$  relaxation restricted to  $|S|$  vertices for any of the hierarchies we consider here).

In our case, this defines a distribution over assignments  $f : S_t \rightarrow \{0, 1\}$  and feasible LP solutions  $\{y_i^f\}_i$  which are integral on  $S_t$ , such that for all  $i \in [n]$  we have  $\mathbb{E}_f[y_i^f] = \|\mathbf{U}_{\{i\}}\|^2$ . Thus, by linearity of expectation, at least one such LP solution has objective value at least  $\sum_i r_i \|\mathbf{U}_{\{i\}}\|^2$ . Moreover, since there are at most  $|S_t|^{t-1}$  assignments in the support of this distribution (each assignment assigns 1 to at most  $t-1$  items), we can enumerate over all such assignments (and corresponding LP solutions) in polynomial time.

## 6.4 Lower Bounds on Integrality Gaps

From the perspective of complexity theory, one can view the various hierarchies of programs as restricted models of computation, with the number of applications of these operators as a resource. This also corresponds naturally to time, as optimizing over the level- $t$  relaxations takes time  $n^{O(t)}$ .

Showing that the integrality gap for a problem remains large after many levels of the hierarchy then corresponds to a strong lower bound, which *unconditionally* (not even assuming  $P \neq NP$ ) rules out a large and general class of algorithms. Also, for some problems where  $NP$ -hardness results are not known, such lower bounds give some evidence of hardness for the problem.

### 6.4.1 Integrality Gaps for Vertex Cover

**Minimum Vertex Cover**, the problem of finding the smallest subset of vertices in a graph such that every edge is incident to some vertex in the subset, is perhaps the most studied problem with regard to integrality gaps. While a simple LP relaxation for the problem gives a factor 2 approximation, it can also be shown that the integrality gap after many levels of the different LP hierarchies remains at least  $2 - \varepsilon$ . However, among the **SDP** hierarchies, lower bounds close to factor 2 are known only in the LS+ hierarchy.

### 6.4.1.1 Viewing Solutions as Local Distributions

The technique most useful for proving lower bounds on LP integrality gaps has been the view of LP solutions as “local distributions” as was stated for the Sherali–Adams hierarchy in Lemma 6.1 (the lemma was stated for Maximum Independent Set, but an identical claim also holds for Minimum Vertex Cover). The LS hierarchy of linear programs does not have such a direct characterization in terms of local distributions, but intuition of probability distributions can still be applied when reasoning about it.

To describe this view for the LS hierarchy, we re-interpret what its conditions mean for a point  $\mathbf{x} \in \mathcal{P}$  which is indeed a convex combination of integer solutions. Such a point  $\mathbf{x}$  is expressible as  $\mathbf{x} = \sum_i \lambda_i \mathbf{z}^{(i)}$  where  $\sum_i \lambda_i = 1$  and  $\forall i. \mathbf{z}^{(i)} \in \mathcal{P} \cap \{0, 1\}^n, \lambda_i \geq 0$ . Then, we can consider a random variable  $\mathbf{z}$  which takes value  $\mathbf{z}^{(i)}$  with probability  $\lambda_i$ . For  $j \in \{1, \dots, n\}$ , the numbers  $x_j$  are then equal to  $\mathbb{P}[z_j = 1]$  i.e. the marginals of this distribution.

To prove that  $\mathbf{x} \in N(\mathcal{P})$ , we then require a matrix  $Y \in \mathbb{R}^{n+1}$  which satisfies the conditions stated in Definition 6.1. For each  $\mathbf{z}^{(i)}$  such a matrix  $Y^{(i)}$  can be given as  $Y^{(i)} = (1, \mathbf{z}^{(i)})(1, \mathbf{z}^{(i)})^T$  where  $(1, \mathbf{z}^{(i)}) \in \{0, 1\}^{n+1} \in \mathbb{R}^{n+1}$ . The matrix  $Y$  for  $\mathbf{x}$  can then be exhibited as  $Y = \sum_i \lambda_i Y^{(i)}$ , where each entry  $Y_{ij} = \mathbb{P}[(z_i = 1) \wedge (z_j = 1)]$ . Arguing that the vector  $Y_i \in \text{cone}(\mathcal{P})$  is then equivalent to arguing that the vector  $\mathbf{x}^{(i,1)} \in \mathbb{R}^n$  defined as

$$x_j^{(i,1)} = Y_{ij}/x_i = \mathbb{P}[z_j = 1 \mid z_i = 1]$$

is in  $\mathcal{P}$ . Similarly,  $Y_0 - Y_i \in \text{cone}(\mathcal{P})$  is equivalent to proving that the vector  $\mathbf{x}^{(i,0)}$  with coordinates  $x_j^{(i,0)} = (Y_{0j} - Y_{ij})/(1 - x_i) = \mathbb{P}[z_j = 1 \mid z_i = 0]$ , is in  $\mathcal{P}$ .

Thus, to prove that a vector  $\mathbf{x}$  of marginal probabilities is in  $N(\mathcal{P})$ , we need to provide a vector of *conditional* probabilities, where the conditioning is on being an arbitrary variable chosen by an *adversary* being 0 or 1. For proving  $\mathbf{x} \in N^t(\mathcal{P})$ , we can think of  $t$ -step game, where at each step we are required to provide the conditional probabilities and the adversary can further condition on one more variable.

### 6.4.1.2 Integrality Gaps in the LS Hierarchy

The study of integrality gaps in this model was initiated by the works of Arora et al. [4, 5]. They showed that the integrality gap remains at least  $2 - \varepsilon$  even after  $\Omega(\log n)$  levels of the hierarchy. Since the integrality gap can be easily shown to be *at most* 2 even for the starting linear relaxation, this showed that even using time  $n^{O(\log n)}$  in the computational model of the LS hierarchy yields no significant improvement. The results were later improved to  $3/2$  for  $\Omega(\log^2 n)$  levels by Tourlakis [57] and to an optimal lower bound of  $2 - \varepsilon$  for  $\Omega(n)$  levels by Schoenebeck et al. [53]. Note that the last result even rules out exponential time algorithms (as  $\Omega(n/\log n)$  levels would correspond to  $2^{\Omega(n)}$  time) in this model.

To produce instances with a large integrality gap, one considers sparse random graphs which have no cycles of size less than  $\Omega(\log n)$  so that any subgraph with  $O(\log n)$  vertices is a tree. One then starts with a solution which has fractional value  $1/2 + \varepsilon$  on every vertex. The move of the adversary then corresponds to selecting a vertex where the solution has a fractional value, and fixing it to 1 or 0 i.e. conditioning it to be in or out of the vertex cover. As long as the adversary conditions on  $O(\log n)$  vertices, the set of conditioned vertices form a tree, restricted to which there is an *actual distribution of vertex covers* with marginal values  $1/2 + \varepsilon$ . Hence, one can use this to produce the required conditional distribution. We remark that this is just an intuition for the proof in [5]. The actual proof proceeds by looking at the duals of the linear programs obtained by the LS hierarchy and involves significantly more work.

The result in [53] for  $\Omega(n)$  levels uses an explicit version of the above intuitive argument together with a more careful use of the sparsity of these graphs. Their techniques also give an integrality gap of  $2 - \varepsilon$  for relaxation of MAX-CUT obtained by  $\Omega(n)$  levels of the LS hierarchy. The results for MAX-CUT also exhibit a separation between linear and semidefinite programs. As shown in Sect. 6.3.1, even a basic semidefinite program at the first level of the SDP hierarchy can be shown to have integrality gap at most  $1/0.878 \approx 1.139$ ; while even the linear programs obtained by  $\Omega(n)$  levels have integrality gap close to 2.

#### 6.4.1.3 Integrality Gaps in the Sherali–Adams Hierarchy

Charikar et al. [16] proved that for any  $\varepsilon > 0$ , there is a  $\delta$  such that the integrality gap of the LP relaxation for Minimum Vertex Cover obtained by  $n^\delta$  levels of the Sherali–Adams hierarchy is at least  $2 - \varepsilon$ . They used an intuition similar to that in [53], where one defines a process to sample vertex covers on trees by including the root with probability  $1/2 + \varepsilon$ ; and including a child with probability 1 if the parent is excluded and  $\varepsilon'$  otherwise. The value of  $\varepsilon'$  is chosen so that the marginal probability for each vertex is  $1/2 + \varepsilon$ . They also defined an extension of this process on graphs which are not trees, to produce local distributions for subsets of size  $n^\delta$ . The number of levels is less than in [53] as the conditions imposed on local distributions by the Sherali–Adams hierarchy are stronger than those imposed by the LS hierarchy.

Using similar techniques, Charikar, Makarychev and Makarychev also showed a gap of  $2 - \varepsilon$  for MAX-CUT after  $n^\delta$  levels. They also extended it show Sherali–Adams integrality gaps for Unique Games and many other problems to which Unique Games can be reduced (see Sect. 6.5 for more on Unique Games).

#### 6.4.1.4 Integrality Gaps for Semidefinite Programs

Integrality gaps for the semidefinite hierarchies have been somewhat harder to prove. It was shown by Goemans and Kleinberg [39] that the integrality gap of an SDP relaxation for Minimum Vertex Cover based on the  $\vartheta$ -function of Lovász

(which is weaker than the relaxation obtained by one application of  $N_+$ ) is at least  $2 - \varepsilon$ . The result was strengthened by Charikar [14], who showed that the same gap holds even when the relaxation is augmented with a subset of the “triangle inequalities” discussed in Sect. 6.3.2. The gap was extended to  $\mathcal{O}(\sqrt{\log n / \log \log n})$  levels of the  $LS_+$  hierarchy by Georgiou et al. [26]. Interestingly, all the these results for  $LS_+$  were proven for the same family of graphs, inspired by a paper of Frankl and Rödl [25]. It is an interesting problem to construct an alternate family which is also an integrality gap instance, or to extend the above results to even  $\mathcal{O}(\log n)$  levels.

Somewhat incomparable to the above results, lower bounds of factor  $7/6$  for  $\mathcal{O}(n)$  levels of the  $LS_+$  hierarchy were obtained by Schoenebeck et al. [52]. These were later strengthened to  $7/6$  for  $\mathcal{O}(n)$  levels of the Lasserre hierarchy by Schoenebeck [51] and  $1.36$  for  $n^{\mathcal{O}(1)}$  levels of Lasserre by Tulsiani [59]. These results also differ from the ones discussed above in that they do not directly exhibit a family of integrality gap instances for vertex cover. Instead, they start with an integrality gap instance for a constraint satisfaction problem, and proceed by using a reduction from the constraint satisfaction problem to vertex cover.

#### 6.4.2 Results for Constraint Satisfaction Problems

For constraint satisfaction problems (CSPs) with three or more variables in each constraint, very strong lower bounds have been shown even for the Lovász–Schrijver and Lasserre semidefinite (and hence also the linear) hierarchies. For these problems one studies how well convex relaxations approximate the maximum number of constraints that can be satisfied by any assignment to the variables. Instances exhibiting a large integrality gap for CSPs are also useful as they can often be transformed to lower bounds for other problems using reductions (see Sect. 6.4.3).

Proofs of integrality gaps for CSPs were significantly influenced by arguments in proof complexity and crucially used an *expansion* property of the problem instances. In proof complexity, expansion arguments were used for proving exponential lower bounds on the size of proofs in the *resolution* proof system to show that a certain *SAT* formula was unsatisfiable.<sup>5</sup> Specifically, they showed that  $\varphi$  is an unsatisfiable formula in  $n$  variables in conjunctive normal form and three variables in each clause, with each set of  $s$  clauses (for say  $s \leq n/1000$ ) involving at least (say)  $3s/2$  variables; then any proof of the unsatisfiability of  $\varphi$  in the resolution proof system must have exponential size (see [9, 10]).

For proving large integrality gaps, one is often interested in showing that such an unsatisfiable *SAT* formula (or instance of some other CSP) “seems highly satisfiable” to a convex relaxation. In the context of the hierarchies, expansion

---

<sup>5</sup>Resolution is the proof system where one uses two clauses of the form  $(\psi_1 \vee x)$  and  $(\psi_2 \vee \neg x)$  to derive  $(\psi_1 \vee \psi_2)$ . Unsatisfiability is proved by deriving the empty clause.

guarantees that any small set of variables is scattered across various clauses and hence the formula restricted to these variables looks highly satisfiable. This intuition is formalized differently for different hierarchies leading to the corresponding bounds.

#### 6.4.2.1 Integrality Gaps for the Lovász–Schrijver Hierarchies

For the LS+ hierarchy, optimal lower bounds of factor  $2^k/(2^k - 1)$  for MAX k-SAT with  $k \geq 5$  and  $\Omega(n)$  levels were shown by Buresh-Oppenheim et al. [13]. They were also the first to use the expansion arguments in the context of Lovász–Schrijver hierarchy. Their results were later extended to the important remaining case of MAX 3-SAT by Alekhnovich et al. [1], who also proved strong lower bounds for approximating Minimum Vertex Cover in hypergraphs.

The arguments for the above results start with a vector with a fractional value for each variable and prove that the vector is in  $N_+^t(\mathcal{P})$  for  $t = \Omega(n)$ . As before, we think of an adversary fixing one of the fractional variables to 1 or 0 (i.e. true or false) at each of  $t$  steps, and one is required to provide a fractional assignment consistent with the fixing which is still in the polytope of feasible solutions. However, at each step, instead of proving that the solutions they provide are in the polytope, they express it as a convex combination of a set  $O$  of fractional solutions, and prove that all the solutions in  $O$  are in the polytope.

The set  $O$  is obtained by fixing *additional* variables at each step to maintain the invariant that if one considers the formula restricted only to the variables which have not been fixed to 0 or 1, then the formula is still expanding (in the sense that a set of  $s$  clauses will contain at least  $3s/2$  unfixed variables). Expansion essentially means that even when  $O(n)$  variables are fixed, most clauses still have a large number of unfixed variables, whose value can be modified suitably to satisfy the constraints of the convex program.

#### 6.4.2.2 Integrality Gaps in the Lasserre Hierarchy

Optimal  $\Omega(n)$  level lower bounds for other constraint satisfaction problems were also proved for the relaxations in the Lasserre hierarchy by Schoenebeck [51] and Tulsiani [59]. Schoenebeck proved the first integrality gaps in the Lasserre hierarchy for the MAX k-XOR problem where each constraint is a linear equation in  $\mathbb{F}_2$  involving  $k$  variables. He showed an optimal integrality gap of  $2 - \varepsilon$  for  $\Omega(n)$  levels of the hierarchy. His techniques were extended in [59] to a large family of CSPs, also showing that for the general problem MAX k-CSP with arbitrary constraints, the integrality gap is at least  $2^k/2k$  after  $\Omega(n)$  levels. The latter result matches, up to a constant factor, the result of Charikar et al. [15], who gave an SDP based approximation algorithm for MAX k-CSP achieving an approximation ratio of  $O(2^k/k)$ .

Schoenebeck's result was based on a significant extension of a technique for creating SDP solutions for **MAX k-XOR**, previously used by Feige and Ofek [24] and Schoenebeck et al. [52]. He showed that for a random instance of **MAX k-XOR**, the SDP relaxation has value 1 (the objective is the maximum fraction of constraints that can be satisfied) while the integer optimum can be easily shown to be at most  $1/2 + \varepsilon$ . He created the SDP solutions for relaxation obtained by  $t$  levels of the Lasserre hierarchy, by creating vectors with one coordinate for each linear form in at most  $2t$  variables over  $\mathbb{F}_2$ . The vector for a partial assignment to the variables then takes value 1 or  $-1$  depending on the parity of the corresponding linear form according to the partial assignment e.g. the vector corresponding to the partial assignment  $(x_1 = 1, x_2 = 1)$  takes value  $(-1)^{x_1} = -1$  in the coordinate for  $x_1$  and  $(-1)^{x_1+x_2} = 1$  in the coordinate for  $x_1 + x_2$ . In addition, the linear equations in the constraints (say  $x_1 + x_2 + x_3 = 1$ ) imply certain additional constraints on the parities ( $(-1)^{x_1+x_2} = -(-1)^{x_3}$  for all assignments). These were imposed by grouping linear forms into equivalence classes if they were related by an equation, and having a single coordinate for each class instead. Expansion was used to prove that it was indeed possible to partition the linear forms consistently.

The above technique was extended in [59] to handle more general constraints, but it still required them to be expressible in some way as linear equations. On the other hand, optimal Sherali–Adams integrality gaps were proved by Georgiou et al. [27] for a much more general class of constraints (which do not have such a linear structure). It remains an interesting open problem to prove the corresponding gaps in the Lasserre hierarchy.

### 6.4.3 Results for Other Problems

A very useful technique in proving integrality gaps for other problems has been the use of *reductions*. This was first used by Khot and Vishnoi [38] in converting integrality gaps for **Unique Games** to those for **Sparsest Cut**. For the purposes of this discussion, we may think of **Unique Games** as simply a CSP with each constraint being a linear equation in two variables modulo a large prime  $p$ . It has been shown to be a very convenient starting point for many reductions. Khot and Vishnoi exhibited an integrality gap for **Unique Games**, and using a reduction to **Sparsest Cut**,<sup>6</sup> proved that the integrality gap is at least  $\Omega((\log \log n)^{1/6})$  for the SDP in Fig. 6.6. The bound was later improved to  $\Omega(\log \log n)$  by Krauthgamer and Rabani [40].

---

<sup>6</sup>The results in [38] were actually for a generalized version of the **Sparsest Cut** problem, where the denominator is not the total number of pairs  $|S||\bar{S}|$  with one vertex in  $S$ , but rather each pair has a different cost associated with it. This is known as the *non-uniform* version of the problem. The result for the uniform version was proven later by Devanur et al. [22].

Their results were extended by Raghavendra and Steurer [50], who showed that the integrality gap remains at least  $\mathcal{O}((\log \log n)^\delta)$  even for the programs obtained by  $\mathcal{O}((\log \log n)^\gamma)$  levels of the mixed hierarchy, for some absolute constants  $\delta, \gamma > 0$ . A similar result was independently obtained by Khot and Saket [37]. The lower bound for the SDP in Fig. 6.6 has been recently improved to  $\mathcal{O}((\log n)^\delta)$  for some small positive constant  $\delta$ , by Cheeger et al. [17]. Note that this still remains far from the best known upper bound of  $O(\sqrt{\log n})$ .

Reductions were also used by Tulsiani [59] to convert integrality gaps for CSPs in the Lasserre hierarchy, to those for Maximum Independent Set, Minimum Vertex Cover and coloring problems. The arguments there involve considering the reductions used for proving the hardness of approximating these problems, and generalizing the proofs of correctness of the reductions to work with vector solutions, instead of integer solutions.

## 6.5 Integrality Gaps and Hardness of Approximation

For constraint satisfaction problems, a very elegant connection between the integrality gaps and the NP-hardness of approximating these problems was exhibited by Raghavendra [49]. He considered a basic semidefinite relaxation for any constraint satisfaction problem, and showed that assuming a conjecture about the hardness of approximating the Unique Games problem, it is NP-hard to achieve a better approximation ratio than the basic SDP.

Recall that Unique Games is a constraint satisfaction problem with each constraint being a linear equation in two variables, modulo a large prime  $p$ . It was conjectured by Khot [35] that for all  $\varepsilon$ , there is a  $p$  such that it is NP-hard to distinguish instances of Unique Games (with equations modulo  $p$ ) in which  $1 - \varepsilon$  fraction of the constraints are satisfiable from those in which at most  $\varepsilon$  fraction are satisfiable. This conjecture, known as the Unique Games Conjecture, has been used as an assumption in a large number of complexity results.

The semidefinite relaxation considered by Raghavendra is best stated in terms of the mixed hierarchy defined in Sect. 6.3.4. For an instance of MAX k-CSP, where each constraint is over  $k$  variables, we consider the program given by the  $k$ th level of the mixed hierarchy (Fig. 6.11). To describe the relaxation when the variables in the CSP are boolean, we introduce real variables  $Y_S$  for all subsets of variables with  $|S| \leq k$ .

$$\begin{aligned} & \text{maximize} && \sum_{i=1}^m \sum_{T_i \subseteq I_i} c_{i,T_i} \cdot Y_{T_i} \\ & \text{subject to} && (Y_S)_{S \subseteq T} \in I_k \quad |T| = k \\ & && M_1(Y) \geq 0 \end{aligned}$$

**Fig. 6.11** Mixed level- $k$  relaxation for a  $k$ -CSP

Let  $m$  be the number of constraints. If  $S_i$  is the set of variables involved in the  $i$ th constraint, then one can find a multilinear polynomial in the variables  $\{x_j\}_{j \in S_i}$  which is 1 when the values of the variables satisfy the constraint and 0 otherwise. Let the polynomial be  $\sum_{T_i \subseteq S_i} c_{i,T_i} \cdot (\prod_{j \in T_i} x_j)$ . We then let the term in the objective function corresponding to the  $i$ th constraint be  $\sum_{T_i \subseteq S_i} c_{i,T_i} Y_{T_i}$ . For example, in the case of an inequality constraint between  $x_1$  and  $x_2$  as in MAX-CUT, the polynomial is  $1 - x_1 - x_2 + 2x_1x_2$  and the term in the objective function is  $Y_0 - Y_{\{1\}} - Y_{\{2\}} + 2Y_{\{1,2\}}$  (where  $Y_\emptyset = 1$ ).

Note that the condition  $M_1(Y) \geq 0$  is equivalent to the existence of vectors  $\mathbf{u}_0, \dots, \mathbf{u}_n$  such that  $Y = \|\mathbf{u}_0\|^2 = 1$  and  $\langle \mathbf{u}_0, \mathbf{u}_i \rangle = \langle \mathbf{u}_i, \mathbf{u}_i \rangle = Y_{\{i\}}$ ,  $\langle \mathbf{u}_i, \mathbf{u}_j \rangle = Y_{\{i,j\}}$  for  $1 \leq i, j \leq n$ . The above program can also be expressed in terms of the variables  $X_{(S,\alpha)}$  for partial assignments as described in Sect. 6.2.2. The formulation in terms of partial assignments can then be generalized for CSPs with non-boolean variables while the above program is specialized to the case of boolean variables.

It is easy to check that for MAX-CUT, the above relaxation is equivalent to the one discussed in Sect. 6.3.1. For MAX-CUT, it was shown by Khot et al. [36] that it is NP-hard to achieve an approximation better than  $1/C_{GW} - \varepsilon$  for any  $\varepsilon > 0$  (assuming the Unique Games Conjecture). This was significantly generalized by Raghavendra who showed that given an instance with integrality gap  $\alpha$  for the above SDP for a given type of constraints, one can convert an algorithm achieving an approximation  $\alpha - \varepsilon$  for the corresponding CSP, to an algorithm for the Unique Games problem. Assuming the Unique Games Conjecture, one then gets that it is NP-hard to achieve an approximation ratio better than the integrality gap of the above program. This points to a very interesting connection between the power of convex relaxations and those of general polynomial time algorithms.

## References

1. Alekhnovich, M., Arora, S., Tourlakis, I.: Towards strong nonapproximability results in the Lovasz-Schrijver hierarchy. In: Gabow, H.N., Fagin, R. (eds.) Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA. pp. 294–303. ACM (2005)
2. Alon, N., Kahale, N.: Approximating the independence number via the theta-function. Math. Program. **80**, 253–264 (1998)
3. Arora, S., Lee, J.R., Naor, A.: Euclidean distortion and the sparsest cut. J. Amer. Math. Soc. **21**(1), 1–21 (2008)
4. Arora, S., Bollobás, B., Lovász, L.: Proving integrality gaps without knowing the linear program. In: Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science, pp. 313–322 (2002)
5. Arora, S., Bollobás, B., Lovász, L., Tourlakis, I.: Proving integrality gaps without knowing the linear program. Theory of Computing **2**(2), 19–51 (2006)
6. Arora, S., Charikar, M., Chlamtac, E.: New approximation guarantee for chromatic number. In: STOC ’06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing, New York, NY, USA. pp. 215–224 (2006)
7. Arora, S., Rao, S., Vazirani, U.: Expander flows and a  $\sqrt{\log n}$ -approximation to sparsest cut. In: Proceedings of the 36th ACM Symposium on Theory of Computing (2004)

8. MohammadHossein Bateni, Moses Charikar, and Venkatesan Guruswami. Maxmin allocation via degree lower-bounded arborescences. In 41st annual ACM symposium on Theory of computing, pages 543–552, New York, NY, USA (2009) ACM.
9. Ben-Sasson, E.: Expansion in Proof Complexity. PhD thesis, Hebrew University (2001)
10. Ben-Sasson, E., Wigderson, A.: Short proofs are narrow: Resolution made simple. *Journal of the ACM* **48**(2) (2001)
11. Blum, A., Karger, D.: An  $\tilde{O}(n^{3/14})$  coloring algorithm for 3-colorable graphs. *Information Processing Letters* **61**(1), 49–53 (1997)
12. Blum, A.: New approximation algorithms for graph coloring. *J. ACM* **41**(3), 470–516 (1994)
13. Buresh-Oppenheim, J., Galesi, N., Hoory, S., Magen, A., Pitassi, T.: Rank bounds and integrality gaps for cutting planes procedures. In: Proceedings of the 44th IEEE Symposium on Foundations of Computer Science, pp. 318–327 (2003)
14. Charikar, M.: On semidefinite programming relaxations for graph coloring and vertex cover. In: Proceedings of the 13th ACM-SIAM Symposium on Discrete Algorithms, pp. 616–620 (2002)
15. Charikar, M., Makarychev, K., Makarychev, Y.: Near-optimal algorithms for maximum constraint satisfaction problems. In: Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms, pp. 62–68 (2007)
16. Charikar, M., Makarychev, K., Makarychev, Y.: Integrality gaps for Sherali-Adams relaxations. In: Proceedings of the 41st ACM Symposium on Theory of Computing (2009)
17. Cheeger, J., Kleiner, B., Naor, A.: A  $(\log n)^{\Omega(1)}$  integrality gap for the sparsest cut SDP. In: FOCS, pp. 555–564 (2009)
18. Chlamtac, E.: Approximation algorithms using hierarchies of semidefinite programming relaxations. In: FOCS, pp. 691–701 (2007)
19. Chlamtac, E., Singh, G.: Improved approximation guarantees through higher levels of SDP hierarchies. In: APPROX-RANDOM, pp. 49–62 (2008)
20. Chvátal, V.: Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics* **4**(4), 305–337 (1973)
21. de la Vega W.F., Kenyon-Mathieu, C.: Linear programming relaxations of maxcut. In: SODA, pp. 53–61 (2007)
22. Devanur, N.R., Khot, S., Saket, R., Vishnoi, N.K.: Integrality gaps for sparsest cut and minimum linear arrangement problems. In: STOC, pp. 537–546 (2006)
23. Feige, U., Kilian, J.: Zero knowledge and the chromatic number. *Journal of Computer and System Sciences* **57**(2), 187–199 (1998)
24. Feige, U., Ofek, E.: Random 3CNF formulas elude the Lovász theta function. Manuscript (2006)
25. Frankl, P., Rodl, V.: Forbidden intersections. *Transactions of the American Mathematical Society* **300**(1), 259–286 (1987)
26. Georgiou, K., Magen, A., Pitassi, T., Tourlakis, I.: Integrality gaps of  $2 - o(1)$  for vertex cover SDPs in the Lovász-Schrijver hierarchy. In: Proceedings of the 48th IEEE Symposium on Foundations of Computer Science, pp. 702–712 (2007)
27. Georgiou, K., Magen, A., Tulsiani, M.: Optimal Sherali-Adams gaps from pairwise independence. In: APPROX-RANDOM (2009)
28. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM* **42**(6), 1115–1145 (1995) Preliminary version in Proc. of STOC’94.
29. Gvozdenović, N., Laurent, M.: The operator  $\psi$  for the chromatic number of a graph. *SIAM J. on Optimization* **19**(2), 572–591 (2008)
30. Halperin, E., Nathaniel, R., Zwick, U.: Coloring k-colorable graphs using relatively small palettes. *J. Algorithms* **45**(1), 72–90 (2002)
31. Ibarra, O.H., Kim, C.E.: Fast approximation for the knapsack and sum of subset problems. *Journal of the ACM* **22**, 463–468 (1975)
32. Kale, S., Seshadhri, C.: Combinatorial approximation algorithms for maxcut using random walks. CoRR, abs/1008.3938 (2010)

33. Karger, D., Motwani, R., Sudan, M.: Approximate graph coloring by semidefinite programming. In: Proceedings of the 35th IEEE Symposium on Foundations of Computer Science, pp. 2–13 (1994)
34. Karlin, A.R., Mathieu, C., Nguyen, C.T.: Integrality gaps of linear and semi-definite programming relaxations for knapsack. In: IPCO (2011)
35. Khot, S.: On the power of unique 2-prover 1-round games. In: Proceedings of the 34th ACM Symposium on Theory of Computing, pp. 767–775 (2002)
36. Khot, S., Kindler, G., Mossel, E., O’Donnell, R.: Optimal inapproximability results for MAX-CUT and other two-variable CSPs? In: Proceedings of the 45th IEEE Symposium on Foundations of Computer Science, pp. 146–154 (2004)
37. Khot, S., Saket, R.: SDP integrality gaps with local  $\ell_1$ -embeddability. In: Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (2009)
38. Khot, S., Vishnoi, N.: The unique games conjecture, integrality gap for cut problems and the embeddability of negative type metrics into  $\ell_1$ . In Proceedings of the 46th IEEE Symposium on Foundations of Computer Science, pp. 53–63 (2005)
39. Kleinberg, J.M., Goemans, M.X.: The Lovász Theta function and a semidefinite programming relaxation of vertex cover. SIAM Journal on Discrete Mathematics **11**, 196–204 (1998)
40. Krauthgamer, R., Rabani, Y.: Improved lower bounds for embeddings into  $L_1$ . SIAM J. Comput. **38**(6), 2487–2498 (2009)
41. Krivelevich, M., Nathaniel, R., Sudakov, B.: Approximating coloring and maximum independent sets in 3-uniform hypergraphs. In: SODA, pp. 327–328 (2001)
42. Lasserre, J.B.: An explicit exact SDP relaxation for nonlinear 0-1 programs. In: Proceedings of the 8th Conference on Integer Programming and Combinatorial Optimization, London, UK, pp. 293–303. Springer-Verlag (2001)
43. Laurent, M.: A comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0-1 programming. Math. Oper. Res. **28**(3), 470–496 (2003)
44. Lawler, E.L.: Fast approximation algorithms for knapsack problems. Math. Oper. Res. **4**(4), 339–356, (1979)
45. Lee, J.R.: On distance scales, embeddings, and efficient relaxations of the cut cone. In: SODA, pp. 92–101 (2005)
46. Lovasz, L.: On the shannon capacity of a graph. Information Theory, IEEE Transactions on **25**(1), 1–7, (1979)
47. Lovasz, L., Schrijver, A.: Cones of matrices and set-functions and 0-1 optimization. SIAM J. on Optimization **1**(12), 166–190 (1991)
48. Magen, A., Moharrami, M.: Robust algorithms for maximum independent set on minor-free graphs based on the Sherali-Adams hierarchy. In: 12th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX), pp. 258–271 (2009)
49. Raghavendra, P.: Optimal algorithms and inapproximability results for every CSP? In: STOC, pp. 245–254 (2008)
50. Raghavendra, P., Steurer, D.: Integrality gaps for strong SDP relaxations of unique games. In: Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (2009)
51. Schoenebeck, G.: Linear level Lasserre lower bounds for certain k-CSPs. In: Proceedings of the 49th IEEE Symposium on Foundations of Computer Science (2008)
52. Schoenebeck, G., Trevisan, L., Tulsiani, M.: A linear round lower bound for Lovász-Schrijver SDP relaxations of vertex cover. In: Proceedings of the 22nd IEEE Conference on Computational Complexity (2007)
53. Schoenebeck, G., Trevisan, L., Tulsiani, M.: Tight integrality gaps for Lovasz-Schrijver LP relaxations of vertex cover and max cut. In: Proceedings of the 39th ACM Symposium on Theory of Computing (2007)
54. Schrijver, A.: A comparison of the Delsarte and Lovász bounds. IEEE Transactions on Information Theory **25**(4), 425–429 (1979)
55. Sherali, H.D., Adams, W.P.: A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. SIAM J. Discrete Math. **3**(3), 411–430 (1990)

56. Soto, J.: Improved analysis of a max cut algorithm based on spectral partitioning. CoRR, abs/0910.0504 (2009)
57. Tourlakis, I.: New lower bounds for vertex cover in the Lovasz-Schrijver hierarchy. In: Proceedings of the 21st IEEE Conference on Computational Complexity (2006)
58. Trevisan, L.: Max cut and the smallest eigenvalue. In: STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing, pp. 263–272, New York, NY, USA (2009)
59. Tulsiani, M.: CSP gaps and reductions in the Lasserre hierarchy. In: Proceedings of the 41st ACM Symposium on Theory of Computing (2009)
60. Zuckerman, D.: Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing* **3**(1), 103–128 (2007)

# Chapter 7

## Relaxations of Combinatorial Problems Via Association Schemes

**Etienne de Klerk, Fernando M. de Oliveira Filho, and Dmitrii V. Pasechnik**

### 7.1 Introduction

Semidefinite programming relaxations of combinatorial problems date back to the work of Lovász [20] from 1979, who proposed a semidefinite programming bound for the size of a maximum stable set in a graph which is now known as the Lovász theta number. More recently, Goemans and Williamson [12] showed how to use semidefinite programming to provide an approximation algorithm for the maximum-cut problem; this algorithm achieves the best known approximation ratio for the problem, which is moreover conjectured to be the best possible ratio under the unique games conjecture, a complexity-theoretical assumption (cf. Khot et al. [15]).

The usual approach to obtaining semidefinite programming relaxations of combinatorial problems has been via binary variable reformulations. For example, the convex hull of the set  $\{xx^T : x \in \{-1, 1\}^n\}$  is approximated by a convex set containing it, namely the *elliptope*

$$\{X \in \mathbb{R}^{n \times n} : X_{ii} = 1 \text{ for } i = 1, \dots, n \text{ and } X \succeq 0\},$$

where  $X \succeq 0$  means that  $X$  is positive semidefinite. Stronger relaxations may be obtained by the use of lift-and-project techniques, which provide hierarchies of

---

E. de Klerk (✉)

Tilburg University, Tilburg, The Netherlands

e-mail: [E.deKlerk@uvt.nl](mailto:E.deKlerk@uvt.nl)

F.M. de Oliveira Filho

Tilburg University, Tilburg, The Netherlands

e-mail: [f.m.deoliveirafilho@uvt.nl](mailto:f.m.deoliveirafilho@uvt.nl)

D.V. Pasechnik

School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore

e-mail: [dima@ntu.edu.sg](mailto:dima@ntu.edu.sg)

tighter and tighter semidefinite programming problems; see e.g. the survey paper by Laurent and Rendl [19] for a description of such methods.

Recently, de Klerk et al. [16] presented a different approach to derive semidefinite programming relaxations for the traveling salesman problem; their approach is based on the theory of association schemes.

An *association scheme* is a set  $\{A_0, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  of 0–1 matrices that satisfy the following properties:

1.  $A_0 = I$  and  $A_0 + \dots + A_d = J$ , where  $J$  is the all-ones matrix.
2.  $A_i^\top \in \{A_0, \dots, A_d\}$  for  $i = 1, \dots, d$ .
3.  $A_i A_j = A_j A_i$  for  $i, j = 0, \dots, d$ .
4.  $A_i A_j \in \text{span}\{A_0, \dots, A_d\}$  for  $i, j = 0, \dots, d$ .

We say that the association scheme  $\{A_0, \dots, A_d\}$  has  $d$  *classes*. Association schemes where all the matrices  $A_i$  are symmetric are called *symmetric association schemes*.

For  $i \in \{1, \dots, d\}$  we let  $i^* \in \{1, \dots, d\}$  be such that  $A_{i^*} = A_i^\top$ .

Now let  $P \in \Pi_n$  be given, where,  $\Pi_n$  denotes the set of  $n \times n$  permutation matrices. Notice that  $\{P^\top A_0 P, \dots, P^\top A_d P\}$  is again an association scheme. In this chapter we show how to obtain semidefinite programming relaxations of the following polytope in  $(\mathbb{R}^{n \times n})^{d+1}$ :

$$\mathcal{H}(A_0, \dots, A_d) := \text{conv} \left\{ (P^\top A_0 P, \dots, P^\top A_d P) : P \in \Pi_n \right\}, \quad (7.1)$$

where  $\{A_0, \dots, A_d\}$  is an association scheme. Thus the vertices of  $\mathcal{H}(A_0, \dots, A_d)$  are all of the form  $\{P^\top A_0 P, \dots, P^\top A_d P\}$  where  $P$  is a permutation matrix.

Many combinatorial problems can be expressed as minimizing a linear function over the polytope  $\mathcal{H}(A_0, \dots, A_d)$ .

As an example, consider the maximum bisection problem: We are given a symmetric matrix  $W \in \mathbb{R}^{2m \times 2m}$  with nonnegative entries, which we see as edge weights on the complete graph  $K_{2m}$  on  $2m$  vertices, and we wish to find a maximum-weight copy of the complete bipartite graph  $K_{m,m}$  in  $K_{2m}$ .

It is a simple matter to check that the  $2m \times 2m$  matrices

$$A_0 = I, A_1 = \begin{pmatrix} 0 & J_m \\ J_m & 0 \end{pmatrix}, \text{ and } A_2 = \begin{pmatrix} J_m - I_m & 0 \\ 0 & J_m - I_m \end{pmatrix}, \quad (7.2)$$

where  $J_m$  is the all-ones  $m \times m$  matrix, form a symmetric association scheme. Notice that  $A_1$  is the adjacency matrix of  $K_{m,m}$ , and that  $A_2$  is the matrix whose nonzero entries mark pairs of vertices of  $K_{m,m}$  that are at distance 2 from each other.

One may now rewrite the maximum bisection problem as the following linear program:

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2} \langle W, X_1 \rangle \\ & (X_0, X_1, X_2) \in \mathcal{H}(A_0, A_1, A_2), \end{aligned}$$

where for matrices  $X, Y \in \mathbb{R}^{n \times n}$  we write  $\langle X, Y \rangle$  for the trace inner product between  $X$  and  $Y$ .

In an analogous way, one may encode other combinatorial problems as linear programs over the polytope (7.1). By deriving semidefinite programming relaxations of the polytope (7.1), we will therefore obtain semidefinite programming relaxations of such combinatorial problems. In particular, we show how to obtain known relaxations for the traveling salesman and the maximum bisection problems, and new relaxations for the cycle covering and maximum  $p$ -section problems.

Finally, our approach may be further generalized by considering coherent configurations instead of association schemes. Coherent configurations are similar to association schemes, but there is no requirement for the matrices to commute. Problems such as the maximum  $(k, l)$ -cut problem can be seen as optimization problems over coherent configurations.

## 7.2 Preliminaries and Notation

All the graphs considered in this chapter are *simple*: They have no loops and no multiple edges.

We denote by  $A^\top$  the transpose of a matrix  $A$ . For complex matrices, we denote by  $A^*$  the conjugate transpose of  $A$ . By  $e \in \mathbb{R}^n$  we denote the all-ones vector.

When we say that a real matrix is positive semidefinite, we mean that it is symmetric and has nonnegative eigenvalues. If a complex matrix is positive semidefinite, then it is Hermitian and has nonnegative eigenvalues. We often use the notation  $A \succeq 0$  to denote that  $A$  is (Hermitian) positive semidefinite.

For matrices  $A$  and  $B \in \mathbb{R}^{n \times n}$ , we denote by  $\langle A, B \rangle$  the *trace inner product* between  $A$  and  $B$ , that is

$$\langle A, B \rangle = \text{trace}(B^\top A) = \sum_{i,j=1}^n A_{ij} B_{ij}.$$

When  $A, B \in \mathbb{C}^{n \times n}$ , the trace inner product is given by

$$\langle A, B \rangle = \text{trace}(B^* A) = \sum_{i,j=1}^n A_{ij} \overline{B_{ij}}.$$

We also need some basic properties of tensor products of vector spaces. For background on tensor products, see the book by Greub [13].

If  $U$  and  $V$  are vector spaces over the same field, we denote by  $U \otimes V$  the tensor product of  $U$  and  $V$ . The space  $U \otimes V$  is spanned by elements of the form  $u \otimes v$  for  $u \in U$  and  $v \in V$ .

When  $U = \mathbb{R}^{m_1 \times n_1}$  and  $V = \mathbb{R}^{m_2 \times n_2}$ , we identify  $U \otimes V$  with the space  $\mathbb{R}^{m_1 m_2 \times n_1 n_2}$ , and for  $A \in \mathbb{R}^{m_1 \times n_1}$  and  $B \in \mathbb{R}^{m_2 \times n_2}$  we identify  $A \otimes B$  with the *Kronecker product* of matrices  $A$  and  $B$ . This is the  $m_1 \times n_1$  block matrix with blocks of size  $m_2 \times n_2$ , block  $(i, j)$  being equal to  $A_{ij}B$ . So  $A \otimes B$  is an  $m_1 m_2 \times n_1 n_2$  matrix. The same definitions hold when  $\mathbb{R}$  is replaced by  $\mathbb{C}$ .

One property of the tensor product of matrices that we will use relates to the trace inner product. Namely, for  $A_1, A_2 \in \mathbb{R}^{m \times m}$  and  $B_1, B_2 \in \mathbb{R}^{n \times n}$  we have

$$\langle A_1 \otimes B_1, A_2 \otimes B_2 \rangle = \langle A_1, A_2 \rangle \langle B_1, B_2 \rangle.$$

The same holds when  $\mathbb{R}$  is replaced by  $\mathbb{C}$ , of course.

Finally,  $\delta_{ij}$  denotes the *Kronecker delta function*, which is equal to 1 if  $i = j$ , and equal to 0 otherwise.

## 7.3 Association Schemes

We defined association schemes in the introduction. Now we give a brief overview of the relevant properties of association schemes that will be of use to us; for background on association schemes we refer the reader to Chap. 12 of the book by Godsil [10], the book of Bannai and Ito [2], Sect. 3.3 in the book by Cameron [5], or the recent survey paper of Martin and Tanaka [22].

### 7.3.1 Sources of Association Schemes

In the introduction we used an association scheme that was related to the complete bipartite graph  $K_{m,m}$ . This idea can be generalized: Association schemes can be obtained from other special classes of graphs as well. In this section, we discuss how to obtain association schemes from so-called *distance-regular graphs*; these schemes will be important in Sect. 7.5 in the encoding of combinatorial problems.

Let  $G = (V, E)$  be a graph of diameter  $d$ . We say that  $G$  is *distance-regular* if for any vertices  $x, y \in V$  and any numbers  $k, l = 0, \dots, d$ , the number of vertices at distance  $k$  from  $x$  and  $l$  from  $y$  depends only on  $k, l$ , and the distance between  $x$  and  $y$ , not depending therefore on the actual choice of  $x$  and  $y$ . A distance-regular graph of diameter 2 is called *strongly regular*. The standard text on distance regular graphs is the book of Brouwer et al. [3].

Now, let  $G$  be a distance-regular graph of diameter  $d$  and label its vertices  $1, \dots, n$ . Then the  $n \times n$  matrices  $A_0, \dots, A_d$  such that

$$A_i(x, y) = \begin{cases} 1 & \text{if } \text{dist}_G(x, y) = i, \\ 0 & \text{otherwise} \end{cases}$$

form a symmetric association scheme with  $d$  classes (cf. Cameron [5], Theorem 3.6), to which we refer as the association scheme of the graph  $G$ . The matrices  $A_i$  above are the *distance matrices* of the graph  $G$ , and  $A_1$  is just the adjacency matrix of  $G$ . The association scheme we used in the introduction to encode the maximum bisection problem was the (symmetric) association scheme of  $K_{m,m}$ .

So from distance-regular graphs one may obtain symmetric association schemes. Examples of distance-regular graphs are the complete bipartite graphs  $K_{m,m}$  and the cycles.

### 7.3.2 Eigenvalues of Association Schemes

Let  $\{A_0, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be an association scheme. The span of the matrices  $A_0, \dots, A_d$  is a matrix  $*$ -algebra, that is, it is a subspace of  $\mathbb{C}^{n \times n}$  closed under matrix multiplication and taking complex conjugate transposes. This algebra is the so-called *Bose–Mesner algebra* of the association scheme, and  $A_0, \dots, A_d$  is an orthogonal basis of this algebra. As we observed in the introduction, if  $P \in \Pi_n$ , then  $\{P^T A_0 P, \dots, P^T A_d P\}$  is also an association scheme, and its Bose–Mesner algebra is isomorphic as an algebra to that of  $\{A_0, \dots, A_d\}$ .

The Bose–Mesner algebra of an association scheme is commutative, and therefore it can be diagonalized, that is, there is a unitary matrix  $U \in \mathbb{C}^{n \times n}$  such that the matrices

$$U^* A_0 U, \dots, U^* A_d U$$

are diagonal. This result implies that the algebra has a basis of idempotents, as we state below.

**Theorem 7.1.** *The Bose–Mesner algebra of an association scheme  $\{A_0, \dots, A_d\}$  has an orthogonal basis  $E_0, \dots, E_d$  of Hermitian idempotents (that is,  $E_i$  is Hermitian and  $E_i^2 = E_i$  for  $i = 0, \dots, d$ ). Moreover,  $E_0 = \frac{1}{n} J$  and  $E_0 + \dots + E_d = I$ .*

Note that the matrices  $E_i$  are positive semidefinite, since they are Hermitian and have zero-one eigenvalues.

We may write the basis  $A_0, \dots, A_d$  of the Bose–Mesner algebra in terms of the basis  $E_0, \dots, E_d$  and vice versa. So there are constants  $p_{ij}$  for  $i, j = 0, \dots, d$  such that

$$A_j = \sum_{i=0}^d p_{ij} E_i$$

for  $j = 0, \dots, d$ . Notice that  $p_{ij}$  is actually the  $i$ -th eigenvalue of the matrix  $A_j$ . The constants  $p_{ij}$  are called the *eigenvalues* of the association scheme  $\{A_0, \dots, A_d\}$ .

We may also write the basis  $E_0, \dots, E_d$  in terms of  $A_0, \dots, A_d$ . Thus there exist constants  $q_{ij}$  for  $i, j = 0, \dots, d$ , the *dual eigenvalues* of the association scheme, such that

$$E_j = \frac{1}{n} \sum_{i=0}^d q_{ij} A_i \tag{7.3}$$

for  $j = 0, \dots, d$ . Our normalization above ensures that  $q_{i0} = 1$  for all  $i = 0, \dots, d$ .

It is customary to define primal and dual matrices of eigenvalues of the association scheme as follows:

$$P = (p_{ij})_{i,j=0}^d \quad \text{and} \quad Q = (q_{ij})_{i,j=0}^d.$$

It is easy to check that  $PQ = nI$ .

For symmetric association schemes, the numbers  $p_{ij}$  and  $q_{ij}$  are all real. In this case, we have the identity

$$\frac{p_{ji}}{m_i} = \frac{q_{ij}}{n \operatorname{trace} E_j}, \quad (7.4)$$

where  $m_i = \langle J, A_i \rangle$ .

Finally, notice that if  $P \in \Pi_n$ , then the eigenvalues and dual eigenvalues of both  $\{A_0, \dots, A_d\}$  and  $\{P^\top A_0 P, \dots, P^\top A_d P\}$  are the same.

There is a closed formula for the eigenvalues and dual eigenvalues of the association scheme of a strongly regular graph. If  $A$  is the adjacency matrix of a strongly regular graph, then  $e$  is an eigenvector of  $A$ , since a strongly regular graph is regular. Eigenvalues associated with eigenvectors orthogonal to  $e$  are said to be the *restricted* eigenvalues of the graph. Every strongly regular graph has exactly two distinct restricted eigenvalues. We then have the following result.

**Theorem 7.2.** *Let  $G$  be a strongly regular graph with  $n$  vertices and of degree  $k$  having restricted eigenvalues  $r$  and  $s$  with  $r > s$ . Suppose that the eigenvalue  $r$  occurs with multiplicity  $f$  and that  $s$  occurs with multiplicity  $g$ . Then with  $n = |V|$  we have*

$$P = \begin{pmatrix} 1 & k & n-k-1 \\ 1 & r & -r-1 \\ 1 & s & -s-1 \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} 1 & f & g \\ 1 & fr/k & gs/k \\ 1 - \frac{f(r+1)}{n-k-1} & -\frac{g(s+1)}{n-k-1} \end{pmatrix}.$$

## 7.4 Semidefinite Programming Relaxations of Association Schemes

Let  $\{A_0, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be an association scheme. We now present a semidefinite programming relaxation of the polytope  $\mathcal{H}(A_0, \dots, A_d)$ , which we defined in (7.1).

Recall that we have expression (7.3) for the dual eigenvalues of an association scheme, and also that each  $E_i$  is positive semidefinite. Since the eigenvalues and dual eigenvalues of an association scheme are the same as the eigenvalues and dual eigenvalues of any permuted version of the association scheme, we see that any  $(X_0, \dots, X_d) \in \mathcal{H}(A_0, \dots, A_d)$  satisfies

$$\begin{aligned} \sum_{i=0}^d X_i &= J, \\ \sum_{i=0}^d q_{ij} X_i &\geq 0 \quad \text{for } j = 1, \dots, d, \\ X_0 &= I, X_i \text{ nonnegative and } X_i^T = X_{i^*} \text{ for } i = 1, \dots, d \end{aligned} \tag{7.5}$$

The theorem below shows that many important constraints are already implied by (7.5).

**Theorem 7.3.** *Let  $\{A_0, \dots, A_d\}$  be an association scheme and suppose  $X_0, \dots, X_d$  satisfy (7.5). Write*

$$Y_j = \frac{1}{n} \sum_{i=0}^d q_{ij} X_i.$$

*Then the following holds:*

1.  $\sum_{j=0}^d Y_j = I$ ;
2.  $Y_j e = 0$  for  $j = 1, \dots, d$ ;
3.  $\sum_{i=0}^d p_{ij} Y_i = X_j$  for  $j = 0, \dots, d$ ;
4.  $X_j e = A_j e = p_0 e$  for  $j = 0, \dots, d$ .

*Proof.* Direct from (7.5) and the properties of the matrices of eigenvalues and dual eigenvalues of the association scheme.  $\square$

In addition, an entire class of linear matrix inequalities is implied by (7.5), as we show next.

**Theorem 7.4.** *Let  $\{A_0, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be an association scheme and suppose that the linear matrix inequality*

$$\sum_{i=0}^d \alpha_i X_i \geq 0, \tag{7.6}$$

*where  $\alpha_0, \dots, \alpha_d$  are given scalars, is satisfied by  $X_i = A_i$ . Then (7.6) may be written as a conic combination of the linear matrix inequalities*

$$\sum_{i=0}^d q_{ij} X_i \geq 0 \quad \text{for } j = 0, \dots, d.$$

*Proof.* We have

$$Y = \sum_{i=0}^d \alpha_i A_i \geq 0.$$

But then since  $Y$  is in the Bose–Mesner algebra of the association scheme, there exist nonnegative numbers  $\beta_0, \dots, \beta_d$  such that

$$Y = \sum_{j=0}^d \beta_j E_j,$$

where  $E_0, \dots, E_d$  is the idempotent basis of the algebra. (Notice that, in fact, the numbers  $\beta_j$  are the eigenvalues of  $Y$ .) Substituting

$$E_j = \frac{1}{n} \sum_{i=0}^d q_{ij} A_i$$

yields

$$\sum_{i=0}^d \alpha_i A_i = \sum_{j=0}^d \frac{\beta_j}{n} \left( \sum_{i=0}^d q_{ij} A_i \right),$$

and since the  $A_i$  are pairwise orthogonal we see that

$$\alpha_i = \sum_{j=0}^d \frac{\beta_j}{n} q_{ij},$$

as we wanted. □

## 7.5 Semidefinite Programming Relaxations of Combinatorial Problems

We now show how to derive semidefinite programming relaxations of many different combinatorial problems from the relaxation for (7.1) which we presented in the previous section.

The basic idea is to restate the combinatorial problem as the problem of finding a maximum/minimum-weight distance-regular subgraph of a weighted complete graph, and then use the fact that the distance matrices of a distance-regular graph give rise to an association scheme. For example, the traveling salesman problem is the problem of finding a minimum-weight Hamiltonian cycle in a graph, and the maximum bisection problem is the problem of finding a maximum-weight copy of  $K_{m,m}$  in the complete graph  $K_{2m}$ .

More precisely, suppose we are given a symmetric matrix  $W \in \mathbb{R}^{n \times n}$  that we see as giving weights to the edges of the complete graph  $K_n$ , and we are also given a distance-regular graph  $H$  on  $n$  vertices. Our problem is to find a maximum-weight copy of  $H$  in  $K_n$ .

Say  $H$  has diameter  $d$ , and let  $A_0, \dots, A_d$  be its distance matrices, so that  $A_1$  is the adjacency matrix of  $H$ . Since  $H$  is distance-regular,  $\{A_0, \dots, A_d\}$  is an association scheme. Our problem can be equivalently stated as the following *quadratic assignment problem*:

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2}\langle W, P^T A_1 P \rangle \\ & \quad P \in \Pi_n. \end{aligned}$$

The problem above can be equivalently stated in terms of the polytope  $\mathcal{H}(A_0, \dots, A_d)$ :

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2}\langle W, X_1 \rangle \\ & \quad (X_0, \dots, X_d) \in \mathcal{H}(A_0, \dots, A_d). \end{aligned}$$

Finally, one may consider relaxation (7.5) for the above problem:

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2}\langle W, X_1 \rangle \\ & \quad \sum_{i=0}^d X_i = J, \\ & \quad \sum_{i=0}^d q_{ij} X_i \geq 0 \quad \text{for } j = 1, \dots, d, \\ & \quad X_0 = I, X_i \text{ nonnegative and symmetric for } i = 1, \dots, d. \end{aligned}$$

This is the approach we employ in the following sections.

### 7.5.1 The Traveling Salesman Problem and the $k$ -Cycle Covering Problem

We are given a symmetric nonnegative matrix  $W \in \mathbb{R}^{n \times n}$ , which we view as giving weights to the edges of the complete graph  $K_n$ . The traveling salesman problem asks us to find a minimum-weight copy of  $C_n$ , the cycle on  $n$  vertices, in the weighted graph  $K_n$ .

Now,  $C_n$  is a distance-regular graph, and its (symmetric) association scheme has  $\lfloor n/2 \rfloor$  classes and is known as the *Lee scheme*, which is also the association scheme of symmetric circulant matrices. For odd  $n$ , the dual eigenvalues of the association scheme of  $C_n$  are given by

$$q_{ij} = 2 \cos(2ij\pi/n) \quad \text{for } i, j = 1, \dots, \lfloor n/2 \rfloor.$$

Moreover,  $q_{i0} = 1$  for  $i = 0, \dots, \lfloor n/2 \rfloor$  and  $q_{0j} = 2$  for  $j = 1, \dots, \lfloor n/2 \rfloor$ .

So, if we consider relaxation (7.5) for the Lee scheme, we obtain the following semidefinite programming relaxation for the traveling salesman problem when  $n$  is odd:

$$\begin{aligned} \text{minimize } & \frac{1}{2}\langle W, X_1 \rangle \\ & I + \sum_{i=1}^d X_i = J, \\ & I + \sum_{i=1}^d \cos(2ij\pi/n)X_i \geq 0 \quad \text{for } j = 1, \dots, \lfloor n/2 \rfloor, \\ & X_i \text{ nonnegative and symmetric for } i = 1, \dots, \lfloor n/2 \rfloor. \end{aligned} \tag{7.7}$$

This is precisely the semidefinite programming relaxation for the traveling salesman problem introduced by de Klerk et al. [16]. One may use Theorem 7.4 to show that this relaxation is tighter than the one given by Cvetković et al. [4]. This fact was already known, but the proof can be greatly simplified by using the more general Theorem 7.4.

A simple modification in the objective function of (7.7) allows us to provide a relaxation for the minimum  $k$ -cycle covering problem. This is the problem of partitioning a weighted complete graph into  $k$  vertex-disjoint cycles of equal length so that the total weight of the cycles is minimized.

This problem has been studied by Goemans and Williamson [11] (see also Manthey [21]), who showed that a 4-approximation algorithm exists when the weights satisfy the triangle inequality.

Now suppose  $n$  is a multiple of  $k$ . One may check that the matrix  $A_k$ , the  $k$ -th distance matrix of the cycle  $C_n$ , is actually the incidence matrix of  $k$  vertex-disjoint cycles of length  $n/k$  each. So, to obtain a relaxation of the  $k$ -cycle covering problem, one only has to replace the objective function of (7.7) by  $\frac{1}{2}\langle W, X_k \rangle$ .

### 7.5.2 The Maximum Bisection Problem

We already considered the maximum bisection problem in the introduction. The problem consists of, given a nonnegative symmetric matrix  $W \in \mathbb{R}^{2m \times 2m}$ , which we see as giving weights to the edges of the complete graph  $K_{2m}$ , finding a maximum-weight copy of the complete bipartite graph  $K_{m,m}$  in  $K_{2m}$ .

The graph  $K_{m,m}$  is a strongly regular graph; the symmetric association scheme induced by it was described in (7.2). Since  $K_{m,m}$  is strongly regular, its dual eigenvalues have a closed formula, as given in Theorem 7.2. Indeed, the restricted eigenvalues of  $K_{m,m}$  are the eigenvalues  $r = 0$  and  $s = -m$  of  $A_1$ , and so the matrices of eigenvalues and dual eigenvalues are given by

$$P = \begin{pmatrix} 1 & m & m-1 \\ 1 & 0 & -1 \\ 1 & -m & m-1 \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} 1 & 2(m-1) & 1 \\ 1 & 0 & -1 \\ 1 & -2 & 1 \end{pmatrix}.$$

Now, by adapting (7.5), we obtain the following relaxation of the maximum bisection problem:

$$\begin{aligned} & \text{maximize } \frac{1}{2}\langle W, X_1 \rangle \\ & I + X_1 + X_2 = J, \\ & (m-1)I - X_2 \geq 0, \\ & I - X_1 + X_2 \geq 0, \\ & X_1, X_2 \text{ nonnegative and symmetric.} \end{aligned} \tag{7.8}$$

Frieze and Jerrum [8] (see also Ye [29]) considered another semidefinite programming relaxation of the maximum bisection problem. Their relaxation is the following:

$$\begin{aligned} & \text{maximize } \frac{1}{4}\langle W, J - X \rangle \\ & X_{ii} = 1 \text{ for } i = 1, \dots, 2m, \\ & Xe = 0, \\ & X \geq 0. \end{aligned} \tag{7.9}$$

To see that this is a relaxation of the problem, notice that  $X = vv^T$  is a feasible solution, where  $v \in \{-1, 1\}^{2m}$  gives a bisection of the vertex set.

Actually, the relaxation we give is equivalent to the one given by Frieze and Jerrum, as we show in the next theorem.

**Theorem 7.5.** *The optimal values of problems (7.8) and (7.9) coincide.*

*Proof.* Given a feasible solution  $X_1, X_2$  of (7.8), set

$$X = I - X_1 + X_2 \geq 0.$$

Using (4) of Theorem 7.3 we have that

$$Xe = e - X_1e + X_2e = e - me + (m-1)e = 0,$$

where we have used that  $A_1e = me$  and  $A_2e = (m-1)e$  for the association scheme of  $K_{m,m}$ . It is also easy to verify that the diagonal entries of  $X$  are all equal to 1 and that

$$\frac{1}{4}\langle W, J - X \rangle = \frac{1}{2}\langle W, X_1 \rangle.$$

Conversely, suppose  $X$  is a feasible solution of (7.9). We claim that

$$X_1 = \frac{1}{2}(J - X) \quad \text{and} \quad X_2 = \frac{1}{2}(J + X) - I$$

is a feasible solution of (7.8) with the same objective value.

Indeed, it is easy to see that  $X_1, X_2$  has the same objective value as  $X$ . To see that it is a feasible solution of (7.8), notice that since  $X$  is positive semidefinite and has diagonal entries equal to 1, all entries of  $X$  have absolute value at most 1, and therefore both  $X_1$  and  $X_2$  are nonnegative. Moreover, it is easy to see that  $X_1$ , and  $X_2$  are symmetric, and that  $I + X_1 + X_2 = J$  and  $I - X_1 + X_2 \succeq 0$ . So we argue that  $(m-1)I - X_2 \succeq 0$ .

Indeed,  $(m-1)I - X_2 = mI - \frac{1}{2}(J + X)$ . Now, the matrix  $M = \frac{1}{2}(J + X)$  is positive semidefinite, and  $e$  is one of its eigenvectors, with associated eigenvalue  $m$ . Since  $M$  has trace  $2m$  and is positive semidefinite, this implies that any other eigenvalue of  $M$  is at most  $m$ . So it follows that  $mI - M \succeq 0$ , as we wanted.  $\square$

### 7.5.3 The Maximum $p$ -Section Problem

Given a nonnegative symmetric matrix  $W \in \mathbb{R}^{pm \times pm}$ , which we see as giving weights to the edges of the complete graph  $K_{pm}$ , our goal is to find a maximum-weight copy of the complete regular  $p$ -partite graph in  $K_{pm}$ .

The complete  $p$ -partite graph is strongly regular, and it generates a symmetric association scheme whose matrix  $Q$  of dual eigenvalues is

$$Q = \begin{pmatrix} 1 & p(m-1) & p-1 \\ 1 & 0 & -1 \\ 1 & -p & p-1 \end{pmatrix}.$$

Then relaxation (7.5) simplifies to

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2}\langle W, X_1 \rangle \\ & I + X_1 + X_2 = J, \\ & (m-1)I - X_2 \succeq 0, \\ & (p-1)I - X_1 + (p-1)X_2 \succeq 0, \\ & X_1, X_2 \text{ nonnegative and symmetric.} \end{aligned}$$

Note that this coincides with relaxation (7.8) for the maximum bisection problem when  $p = 2$ . A different semidefinite programming relaxation for the maximum  $p$ -section problem was proposed by Andersson [1].

## 7.6 Semidefinite Programming Relaxations of Coherent Configurations

A *coherent configuration* is a set  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  of 0–1 matrices with the following properties:

1. There is a set  $N \subseteq \{1, \dots, d\}$  such that  $\sum_{i \in N} A_i = I$  and  $A_1 + \dots + A_d = J$ .
2.  $A_i^\top \in \{A_1, \dots, A_d\}$  for  $i = 1, \dots, d$ .
3.  $A_i A_j \in \text{span}\{A_1, \dots, A_d\}$  for  $i, j = 1, \dots, d$ .

Thus association schemes are commutative coherent configurations.

GAP [9] software for computing with coherent configurations is described in Pasechnik and Kini [24].

Let  $\{A_1, \dots, A_d\}$  be a coherent configuration. For  $i \in \{1, \dots, d\}$  we let  $i^* \in \{1, \dots, d\}$  be such that  $A_{i^*} = A_i^\top$ , as before. We also write

$$m_i = \langle J, A_i \rangle = \langle A_i, A_i \rangle$$

for  $i = 1, \dots, d$ .

Matrices  $A_1, \dots, A_d$  generate a matrix  $*$ -algebra containing the identity. Note that this algebra will not be commutative in general. It is still possible to block-diagonalize the algebra of a coherent configuration, as was shown by Wedderburn [28]. To precisely state Wedderburn's result we first introduce some notation.

Let  $\mathcal{A}, \mathcal{B} \subseteq \mathbb{C}^{n \times n}$ . We write

$$\mathcal{A} \oplus \mathcal{B} = \left\{ \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} : A \in \mathcal{A} \text{ and } B \in \mathcal{B} \right\}.$$

For  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  and for any positive integer  $r$  we write

$$r \odot \mathcal{A} = \{I_r \otimes A : A \in \mathcal{A}\},$$

where  $I_r$  is the  $r \times r$  identity matrix, so that  $I_r \otimes A$  is an  $rn \times rn$  matrix with  $r$  repeated diagonal blocks equal to  $A$  and zeros everywhere else. Finally, let  $U \in \mathbb{C}^{n \times n}$ . Then for  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  we write

$$U^* \mathcal{A} U = \{U^* A U : A \in \mathcal{A}\}.$$

Wedderburn has shown the following theorem.

**Theorem 7.6.** *Let  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  be a matrix  $*$ -algebra containing the identity. Then there is a unitary matrix  $U \in \mathbb{C}^{n \times n}$  and positive integers  $r_1, \dots, r_k$  and  $s_1, \dots, s_k$  such that*

$$U^* \mathcal{A} U = \bigoplus_{i=1}^k r_i \odot \mathbb{C}^{s_i \times s_i}.$$

Notice that, if  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  is a coherent configuration and  $P \in \Pi_n$ , then  $\{P^\top A_1 P, \dots, P^\top A_d P\}$  is also a coherent configuration. As before, our goal is to provide a semidefinite programming relaxation of the polytope

$$\mathcal{H}(A_1, \dots, A_d) := \text{conv}\{(P^\top A_1 P, \dots, P^\top A_d P) : P \in \Pi_n\}. \quad (7.10)$$

To provide a relaxation for this set, we use the following theorem, which describes a linear matrix inequality satisfied by a coherent configuration.

**Theorem 7.7.** *Let  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be a coherent configuration. Then*

$$\sum_{i=1}^d m_i^{-1} A_i \otimes A_i \succeq 0. \quad (7.11)$$

We prove the theorem in a moment; first we use it to describe our semidefinite programming relaxation of (7.10). Let  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be a coherent configuration. From Theorem 7.7, it is clear that for  $(X_1, \dots, X_d) \in \mathcal{H}(A_1, \dots, A_d)$  we have

$$\sum_{i=1}^d m_i^{-1} A_i \otimes X_i \succeq 0. \quad (7.12)$$

Let  $N \subseteq \{1, \dots, d\}$  be such that  $\sum_{i \in N} A_i = I$ . Then any  $(X_1, \dots, X_d) \in \mathcal{H}(A_1, \dots, A_d)$  satisfies

$$\begin{aligned} \sum_{i \in N} X_i &= I, \\ \sum_{i=1}^d X_i &= J, \\ \sum_{i=1}^d m_i^{-1} A_i \otimes X_i &\succeq 0, \\ \langle J, X_i \rangle &= m_i, X_{i^*} = X_i^\top, \text{ and } X_i \geq 0 \text{ for } i = 1, \dots, d. \end{aligned} \quad (7.13)$$

We observe that, in the linear matrix inequality (7.12), one may replace the matrices  $A_i$  from the coherent configuration by their block-diagonalizations, obtaining an equivalent constraint. Also, repeated blocks may be eliminated: Only one copy of each set of repeated blocks is necessary. So, from Wedderburn's theorem we see that in constraint (7.12) it is possible to replace the matrices  $A_i$ , which are  $n \times n$  matrices, by matrices whose dimensions depend only on the dimension of the algebra generated by  $A_1, \dots, A_d$ , which is equal to  $d$ .

In a similar way to Theorem 7.4, the linear matrix inequality (7.12) implies an entire class of linear matrix inequalities that are valid for (7.10), as we will see after the proof of Theorem 7.7.

To prove Theorem 7.7 we need the following theorem that holds in the more general context of  $C^*$ -algebras. We give a proof here for the case of matrix  $*$ -algebras to make our presentation self-contained.

**Theorem 7.8.** *Let  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  be a matrix  $*$ -algebra containing the identity and let  $X \in \mathbb{C}^{n \times n}$  be a positive semidefinite matrix. Then the orthogonal projection of  $X$  onto  $\mathcal{A}$  is also positive semidefinite.*

*Proof.* From Wedderburn's theorem (Theorem 7.6) we may assume that

$$\mathcal{A} = \bigoplus_{i=1}^k r_i \odot \mathbb{C}^{s_i \times s_i}. \quad (7.14)$$

Assume that a Hermitian positive semidefinite  $X \in \mathbb{C}^{n \times n}$  is given. For  $i = 1, \dots, k$  and  $j = 1, \dots, r_i$ , let  $X^{(ij)}$  be the  $s_i \times s_i$  diagonal block of  $X$  corresponding to the  $j$ -th copy of  $\mathbb{C}^{s_i \times s_i}$  in the decomposition of  $\mathcal{A}$  given in (7.14).

Since  $X$  is positive semidefinite, every matrix  $X^{(ij)}$  as defined above is also positive semidefinite. Moreover, the projection of  $X$  onto  $\mathcal{A}$  is explicitly given as

$$\bigoplus_{i=1}^k I_{r_i} \otimes \left( r_i^{-1} \sum_{j=1}^{r_i} X^{(ij)} \right),$$

which is clearly also positive semidefinite.  $\square$

We now give a proof of Theorem 7.7.

*Proof (of Theorem 7.7).* Consider the matrix

$$X = \sum_{k,l=1}^n E_{kl} \otimes E_{kl},$$

where  $E_{kl}$  is the  $n \times n$  matrix with 1 in position  $(k, l)$  and 0 everywhere else. Notice that  $X$  is positive semidefinite, since its rows and columns can be permuted to obtain the matrix  $J_n \otimes E_{11}$ .

Now notice that, since  $\{A_1, \dots, A_d\}$  is a coherent configuration, the matrices

$$(m_i m_j)^{-1/2} A_i \otimes A_j \quad \text{for } i, j = 1, \dots, d \quad (7.15)$$

span a matrix  $*$ -algebra that contains the identity. Actually, the matrices above form an orthonormal basis of this algebra.

By Theorem 7.8, the projection of  $X$  onto the algebra spanned by the matrices in (7.15) is a positive semidefinite matrix. We show now that this projection is exactly the matrix in (7.11). Indeed, the projection is explicitly given by

$$\begin{aligned} & \sum_{i,j=1}^d (m_i m_j)^{-1} A_i \otimes A_j \left\langle \sum_{k,l=1}^d E_{kl} \otimes E_{kl}, A_i \otimes A_j \right\rangle \\ &= \sum_{i,j=1}^d (m_i m_j)^{-1} A_i \otimes A_j \sum_{k,l=1}^d \langle E_{kl}, A_i \rangle \langle E_{kl}, A_j \rangle. \end{aligned}$$

Notice that, for  $i, j \in \{1, \dots, d\}$  with  $i \neq j$  we have  $\langle E_{kl}, A_i \rangle \langle E_{kl}, A_j \rangle = 0$ , because the matrices  $A_1, \dots, A_d$  have disjoint supports. So the above expression simplifies to

$$\sum_{i=1}^d m_i^{-2} A_i \otimes A_i \sum_{k,l=1}^n \langle E_{kl}, A_i \rangle^2 = \sum_{i=1}^d m_i^{-1} A_i \otimes A_i,$$

which is exactly the matrix in (7.11), as we wanted.  $\square$

Theorem 7.8 also suggests a whole class of linear matrix inequalities that are valid for (7.10). Indeed, let  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be a coherent configuration. Given a positive semidefinite matrix  $Y \in \mathbb{R}^{n \times n}$ , Theorem 7.8 implies that the projection of  $Y$  onto the algebra spanned by  $A_1, \dots, A_d$  is positive semidefinite, that is

$$\sum_{i=1}^d m_i^{-1} \langle Y, A_i \rangle A_i \geq 0.$$

Thus we see that any  $(X_1, \dots, X_d) \in \mathcal{H}(A_1, \dots, A_d)$  satisfies

$$\sum_{i=1}^d m_i^{-1} \langle Y, A_i \rangle X_i \geq 0.$$

All these infinitely many linear matrix inequalities are already implied by (7.12), as we show next.

**Theorem 7.9.** *Let  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be a coherent configuration. Suppose matrices  $X_1, \dots, X_d \in \mathbb{R}^{n \times n}$  with  $X_{i^*} = X_i^\top$  ( $i = 1, \dots, d$ ) satisfy (7.12). Then*

$$\sum_{i=1}^d m_i^{-1} \langle Y, A_i \rangle X_i \geq 0, \tag{7.16}$$

where  $Y \in \mathbb{R}^{n \times n}$  is any positive semidefinite matrix.

*Proof.* Let  $M$  be the matrix in (7.16). It is clear that  $M$  is symmetric. To see that it is actually positive semidefinite, let  $Z \in \mathbb{R}^{n \times n}$  be a positive semidefinite matrix. We show that  $\langle Z, M \rangle \geq 0$ .

Indeed, we have

$$\langle Z, M \rangle = \sum_{i=1}^d m_i^{-1} \langle Y, A_i \rangle \langle Z, X_i \rangle.$$

Now, notice that the matrix  $Y \otimes Z$  is positive semidefinite, since both  $Y$  and  $Z$  are positive semidefinite. Since  $X_1, \dots, X_d$  satisfy (7.12) we have that

$$\begin{aligned} 0 &\leq \left\langle Y \otimes Z, \sum_{i=1}^d m_i^{-1} A_i \otimes X_i \right\rangle \\ &= \sum_{i=1}^d m_i^{-1} \langle Y, A_i \rangle \langle Z, X_i \rangle \\ &= \langle Z, M \rangle, \end{aligned}$$

as required.  $\square$

We also have the following theorem, which is analogous to Theorem 7.4.

**Theorem 7.10.** *Let  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$  be a coherent configuration and suppose that the linear matrix inequality*

$$\sum_{i=1}^d \alpha_i X_i \succeq 0, \quad (7.17)$$

where  $\alpha_1, \dots, \alpha_d$  are scalars such that  $\alpha_{i^*} = \alpha_i$  for  $i = 1, \dots, d$ , is satisfied by  $X_i = A_i$ .

If matrices  $X_1, \dots, X_d \in \mathbb{R}^{n \times n}$  such that  $X_{i^*} = X_i^\top$  for  $i = 1, \dots, d$  satisfy (7.12), then  $X_1, \dots, X_d$  also satisfy (7.17).

*Proof.* We claim that there is a positive semidefinite matrix  $Y \in \mathbb{R}^{n \times n}$  such that

$$\langle Y, A_i \rangle = \alpha_i m_i \quad \text{for } i = 1, \dots, d.$$

Indeed, from Farkas' lemma we know that either there exists such a matrix  $Y$ , or there are numbers  $\beta_1, \dots, \beta_d$  such that

$$\sum_{i=1}^d \beta_i A_i \succeq 0 \quad \text{and} \quad \sum_{i=1}^d \alpha_i \beta_i m_i < 0.$$

Now, since (7.17) is satisfied by  $X_i = A_i$ , we know that

$$\begin{aligned} 0 &\leq \left\langle \sum_{i=1}^d \alpha_i A_i, \sum_{j=1}^d \beta_j A_j \right\rangle \\ &= \sum_{i,j=1}^d \alpha_i \beta_j \langle A_i, A_j \rangle \\ &= \sum_{i=1}^d \alpha_i \beta_i m_i \\ &< 0, \end{aligned}$$

a contradiction. Here, we used the fact that  $\langle A_i, A_j \rangle = \delta_{ij} m_i$  for  $i, j = 1, \dots, d$ . So there must be a matrix  $Y$  as claimed.

Now, if  $Y$  is given as in our claim, then we have that

$$\sum_{i=1}^d m_i^{-1} \langle Y, A_i \rangle X_i = \sum_{i=1}^d \alpha_i X_i,$$

and since  $X_1, \dots, X_d$  satisfy (7.12), from Theorem 7.9 we see that the matrix above is positive semidefinite, as we wanted.  $\square$

Finally, we observe that for an association scheme  $\{A_0, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$ , relaxation (7.13) is equivalent to (7.5).

To see this, it suffices to show that if  $\{A_0, \dots, A_d\}$  is an association scheme, then the constraint

$$\sum_{i=0}^d m_i^{-1} A_i \otimes X_i \geq 0$$

is equivalent to the constraints

$$\sum_{i=0}^d q_{ij} X_i \geq 0 \quad \text{for } j = 0, \dots, d. \quad (7.18)$$

Indeed, since  $\{A_0, \dots, A_d\}$  is an association scheme, the matrices  $A_0, \dots, A_d$  can be simultaneously diagonalized via a unitary transformation. Then we may write

$$A_i = \sum_{j=0}^d p_{ji} E_j,$$

where the matrices  $E_0, \dots, E_d$  are 0–1 diagonal matrices that sum to the identity. So we see that the linear matrix inequality

$$\sum_{i=0}^d m_i^{-1} \sum_{j=0}^d p_{ji} E_j \otimes X_i = \sum_{i=0}^d m_i^{-1} A_i \otimes X_i \geq 0$$

is equivalent to the linear matrix inequalities

$$\sum_{i=0}^d m_i^{-1} p_{ji} X_i \geq 0 \quad \text{for } j = 0, \dots, d,$$

and by using (7.4) we get that the above constraints are equivalent to (7.18), as we wanted.

### 7.6.1 Relation to a Relaxation of the Quadratic Assignment Problem

Given symmetric matrices  $A$  and  $W \in \mathbb{R}^{n \times n}$ , a simplified version of the quadratic assignment problem is as follows:

$$\begin{aligned} & \text{maximize } \langle W, P^T A P \rangle \\ & P \in \Pi_n. \end{aligned}$$

Povh and Rendl [25] proposed the following semidefinite programming relaxation for this problem:

$$\begin{aligned} & \text{maximize } \langle A \otimes W, Y \rangle \\ & \langle I \otimes E_{jj}, Y \rangle = \langle E_{jj} \otimes I, Y \rangle = 1 \quad \text{for } j = 1, \dots, n, \\ & \langle I \otimes (J - I) + (J - I) \otimes I, Y \rangle = 0, \\ & \langle J \otimes J, Y \rangle = n^2, \\ & Y \geq 0 \text{ and } Y \succeq 0, \end{aligned} \tag{7.19}$$

where  $E_{jj}$  is the matrix with 1 in position  $(j, j)$  and 0 everywhere else. This relaxation is in fact equivalent to an earlier one due to Zhao et al. [30]; see also the Chap. 27 of this Handbook for more details on semidefinite programming relaxations of quadratic assignment and related problems.

When the matrix  $A$  belongs to the span of a coherent configuration  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$ , then one may use (7.13) to provide a semidefinite programming relaxation of the quadratic assignment problem as follows

$$\begin{aligned} & \text{maximize } \left\langle W, \sum_{i=1}^d m_i^{-1} \langle A, A_i \rangle X_i \right\rangle \\ & \sum_{i \in N} X_i = I, \\ & \sum_{i=1}^d X_i = J, \\ & \sum_{i=1}^d m_i^{-1} A_i \otimes X_i \succeq 0, \\ & \langle J, X_i \rangle = m_i, X_{i^*} = X_i^T, \text{ and } X_i \geq 0 \text{ for } i = 1, \dots, d. \end{aligned} \tag{7.20}$$

Actually, when  $A$  belongs to the span of a coherent configuration, relaxations (7.19) and (7.20) are equivalent. This result is essentially due to de Klerk and Sotirov [17], but we give a proof for the sake of completeness. The proof uses the technique of symmetry reduction of invariant semidefinite programs, that is discussed in detail in the Chap. 9 of this Handbook.

**Theorem 7.11 (cf. de Klerk and Sotirov [17]).** *If  $A$  belongs to the span of a coherent configuration  $\{A_1, \dots, A_d\} \subseteq \mathbb{R}^{n \times n}$ , then the optimal values of (7.19) and (7.20) coincide.*

*Proof.* We first show that the optimal value of (7.19) is at most that of (7.20).

To this end, let  $Y \in \mathbb{R}^{n \times n} \otimes \mathbb{R}^{n \times n}$  be a feasible solution of (7.19). Write  $\mathcal{A} = \text{span}\{A_1, \dots, A_d\}$  and consider the matrices

$$m_i^{-1/2} A_i \otimes E_{kl} \quad \text{for } i = 1, \dots, d \text{ and } k, l = 1, \dots, n,$$

where  $E_{ij} \in \mathbb{R}^{n \times n}$  is the matrix with 1 at position  $(i, j)$  and 0 everywhere else. The matrices above form an orthonormal basis of the matrix  $*$ -algebra  $\mathcal{A} \otimes \mathbb{C}^{n \times n}$ . From Theorem 7.8, we know that the orthogonal projection of  $Y$  onto this algebra is a positive semidefinite matrix. In view of the basis given above for the algebra, this projection may be explicitly written as

$$\begin{aligned} \bar{Y} &= \sum_{i=1}^d \sum_{k,l=1}^n m_i^{-1} \langle A_i \otimes E_{kl}, Y \rangle A_i \otimes E_{kl} \\ &= \sum_{i=1}^d m_i^{-1} A_i \otimes X_i \geq 0, \end{aligned}$$

where the  $X_i \in \mathbb{R}^{n \times n}$  ( $i = 1, \dots, d$ ) are suitable matrices. Note that  $\bar{Y}$  is a nonnegative matrix, since  $Y$  is nonnegative.

We now show that  $X_1, \dots, X_d$  is a feasible solution of (7.20). Indeed, it is immediate that  $X_i^* = X_i^\top$  for  $i = 1, \dots, d$ , since  $\bar{Y}^\top = \bar{Y}$ . Moreover, since the  $A_i$  are 0–1 matrices with disjoint supports and  $\bar{Y}$  is nonnegative, we also see that the  $X_i$ 's are nonnegative.

To see that  $\sum_{i \in N} X_i = I$ , notice that for all  $j = 1, \dots, n$ , we have  $I \otimes E_{jj} \in \mathcal{A} \otimes \mathbb{C}^{n \times n}$ . Then, since  $Y$  is feasible for (7.19) and since  $\{A_1, \dots, A_d\}$  is a coherent configuration, we have for  $j = 1, \dots, n$  that

$$1 = \langle I \otimes E_{jj}, \bar{Y} \rangle = \sum_{i=1}^d m_i^{-1} \langle I, A_i \rangle \langle E_{jj}, X_i \rangle = \sum_{i \in N} \langle E_{jj}, X_i \rangle. \quad (7.21)$$

Now, since  $Y$  is feasible for (7.19) we also have that

$$\begin{aligned} 0 &= \langle I \otimes (J - I) + (J - I) \otimes I, Y \rangle \\ &= \langle I \otimes (J - I) + (J - I) \otimes I, \bar{Y} \rangle \\ &= \sum_{i=1}^d m_i^{-1} (\langle I, A_i \rangle \langle J - I, X_i \rangle + \langle J - I, A_i \rangle \langle I, X_i \rangle). \end{aligned} \quad (7.22)$$

Since each  $X_i$  is nonnegative, this implies that whenever  $i \in N$ , all off-diagonal entries of  $X_i$  are equal to zero. Together with (7.21) this implies that  $\sum_{i \in N} X_i = I$ .

We now claim that for  $i \in N$  one has  $\langle J, X_i \rangle = m_i$ . Indeed, since  $\langle E_{jj} \otimes I, Y \rangle = 1$  for  $j = 1, \dots, n$ , for  $i \in N$  we have that  $\langle A_i \otimes I, Y \rangle = m_i$ . But then we have that

$$m_i = \langle A_i \otimes I, \bar{Y} \rangle = \sum_{j=1}^d m_j^{-1} \langle A_i, A_j \rangle \langle I, X_j \rangle = \langle I, X_i \rangle.$$

Since  $\sum_{i \in N} X_i = I$  and since each  $X_i$  is nonnegative, we must have that  $\langle J, X_i \rangle = \langle I, X_i \rangle = m_i$  for  $i \in N$ , as required.

Now we show that in fact  $\langle J, X_i \rangle = m_i$  for  $i = 1, \dots, d$ . To see this, notice that the matrix

$$\sum_{i=1}^d m_i^{-1} \langle J, X_i \rangle A_i = (I \otimes e)^T \left( \sum_{i=1}^d m_i^{-1} A_i \otimes X_i \right) (I \otimes e)$$

is positive semidefinite. But since  $\langle J, X_i \rangle = m_i$  for  $i \in N$ , the diagonal entries of this matrix are all equal to 1. So since this is a positive semidefinite matrix, it must be that all of its entries have absolute value at most 1. But then, since  $\{A_1, \dots, A_d\}$  is a coherent configuration, and since each  $X_i$  is nonnegative, it must be that  $\langle J, X_i \rangle \leq m_i$  for  $i = 1, \dots, n$ .

Now, we know that

$$n^2 = \langle J \otimes J, Y \rangle = \langle J \otimes J, \bar{Y} \rangle = \sum_{i=1}^d \langle J, X_i \rangle,$$

and it follows that  $\langle J, X_i \rangle = m_i$  for  $i = 1, \dots, n$ .

To prove that  $X_1, \dots, X_d$  is feasible for (7.20), we still have to show that  $X_1 + \dots + X_d = J$ . To this end, notice that, since the  $X_i$  are nonnegative, (7.22) implies that every  $X_i$  with  $i \notin N$  has only zeros on the diagonal. So the matrix  $X_1 + \dots + X_d$  has diagonal entries equal to 1 and then, since it is positive semidefinite (as  $\bar{Y}$  is positive semidefinite), and since

$$\sum_{i=1}^d \langle J, X_i \rangle = n^2,$$

we see immediately that  $X_1 + \dots + X_d = J$ , as we wanted.

So  $X_1, \dots, X_d$  is feasible for (7.20). Finally, the objective value of  $X_1, \dots, X_d$  coincides with that of  $Y$ , and we are done.

To see now that the optimal value of (7.20) is at most that of (7.19), one has to show that, given a solution  $X_1, \dots, X_d$  of (7.20), the matrix

$$Y = \sum_{i=1}^d m_i^{-1} A_i \otimes X_i$$

is a feasible solution of (7.19) with the same objective value. This is straight-forward and left to the reader.  $\square$

### 7.6.2 Coherent Configurations and Combinatorial Optimization Problems

In Sect. 7.5 we saw how the problem of finding a maximum/minimum-weight subgraph  $H$  of a complete graph can be modeled with the help of association schemes when the graph  $H$  is distance-regular. For a general graph  $H$ , we show now how to model this problem as a semidefinite programming problem with the help of coherent configurations.

Let  $H = (V, E)$  be a graph, with  $V = \{1, \dots, n\}$ . An *automorphism* of  $H$  is a permutation  $\pi: V \rightarrow V$  that preserves the adjacency relation. The set of all automorphisms of  $H$ , denoted by  $\text{Aut}(H)$ , is a group under the operation of function composition.

For  $(x, y), (x', y') \in V \times V$ , we write

$$(x, y) \sim (x', y')$$

if there is  $\pi \in \text{Aut}(H)$  such that  $\pi(x) = x'$  and  $\pi(y) = y'$ . Relation  $\sim$  is an equivalence relation that partitions  $V \times V$  into equivalence classes called *orbits*. Say  $V \times V / \sim = \{C_1, \dots, C_d\}$  and consider the matrices  $A_1, \dots, A_d \in \mathbb{R}^{n \times n}$  such that

$$A_i(x, y) = \begin{cases} 1 & \text{if } (x, y) \in C_i, \\ 0 & \text{otherwise.} \end{cases}$$

We claim that  $\{A_1, \dots, A_d\}$  is a coherent configuration.

Indeed, it is easy to see that  $\{A_1, \dots, A_d\}$  satisfies items (1)–(2) in the definition of a coherent configuration. We show that item (3) is also satisfied, that is, that the span of  $\{A_1, \dots, A_d\}$  is a matrix  $*$ -algebra.

To see this, associate to each permutation  $\pi \in \text{Aut}(H)$  a permutation matrix  $P_\pi \in \mathbb{R}^{n \times n}$  in the obvious way. Then the set of matrices

$$\{A \in \mathbb{R}^{n \times n} : AP_\pi = P_\pi A \text{ for } \pi \in \text{Aut}(H)\}$$

is exactly the span of  $\{A_1, \dots, A_d\}$ , and it can be then easily seen that this span is also a matrix  $*$ -algebra. So  $\{A_1, \dots, A_d\}$  is a coherent configuration.

Now suppose we are given a symmetric matrix  $W \in \mathbb{R}^{n \times n}$  which we see as giving weights to the edges of the complete graph  $K_n$ , and suppose we are asked to find a maximum-weight copy of  $H$  in  $K_n$ . Let  $A$  be the adjacency matrix of  $H$ . Then we may write this problem equivalently as

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2} \langle W, P^T AP \rangle \\ & P \in \Pi_n. \end{aligned}$$

It can be easily seen that, for some subset  $M \subseteq \{1, \dots, d\}$ , one has

$$A = \sum_{i \in M} A_i.$$

So the problem above can be rewritten as

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2} \left\langle W, \sum_{i \in M} X_i \right\rangle \\ & (X_1, \dots, X_d) \in \mathcal{H}(A_1, \dots, A_d). \end{aligned}$$

Finally, when we consider relaxation (7.13) for this problem, we obtain

$$\begin{aligned} & \text{maximize} \quad \frac{1}{2} \left\langle W, \sum_{i \in M} X_i \right\rangle \\ & \sum_{i \in N} X_i = I, \\ & \sum_{i=1}^d X_i = J, \\ & \sum_{i=1}^d m_i^{-1} A_i \otimes X_i \geq 0, \\ & \langle J, X_i \rangle = m_i, X_{i^*} = X_i^T, \text{ and } X_i \geq 0 \text{ for } i = 1, \dots, d. \end{aligned}$$

In our discussion it was not in any moment necessary to work with the full automorphism group of  $H$ ; any subgroup of this group will do. In any case, the larger the group one uses, the smaller the matrices  $A_i$  can be made to be after the block-diagonalization.

If the automorphism group of  $H$  acts transitively on its vertex set, then it is possible to obtain stronger semidefinite programming relaxations by using a stabilizer subgroup instead of the full automorphism group; see de Klerk and

Sotirov [18]. A similar idea was used by Schrijver [27] earlier to obtain improved semidefinite programming bounds for binary code sizes.

### 7.6.3 The Maximum $(k,l)$ -Cut Problem

The maximum  $(k,l)$ -cut problem is a generalization of the maximum bisection problem which consists of the following: Given a symmetric nonnegative matrix  $W \in \mathbb{R}^{n \times n}$ , which we see as giving weights to the edges of the complete graph  $K_n$ , where  $n = k + l$ , find a maximum-weight copy of  $K_{k,l}$  in  $K_n$ .

We may use our semidefinite programming relaxation of (7.10) to give a relaxation of the maximum  $(k,l)$ -cut problem by considering the following coherent configuration associated with the complete bipartite graph  $K_{k,l}$ :

$$\begin{aligned} A_1 &= \begin{pmatrix} I_k & 0_{k \times l} \\ 0_{l \times k} & 0_{l \times l} \end{pmatrix}, \quad A_2 = \begin{pmatrix} J_k - I_k & 0_{k \times l} \\ 0_{l \times k} & 0_{l \times l} \end{pmatrix}, \quad A_3 = \begin{pmatrix} 0_{k \times k} & J_{k \times l} \\ 0_{l \times k} & 0_{l \times l} \end{pmatrix}, \\ A_4 &= \begin{pmatrix} 0_{k \times k} & 0_{k \times l} \\ J_{l \times k} & 0_{l \times l} \end{pmatrix}, \quad A_5 = \begin{pmatrix} 0_{k \times k} & 0_{k \times l} \\ 0_{l \times k} & I_l \end{pmatrix}, \quad A_6 = \begin{pmatrix} 0_{k \times k} & 0_{k \times l} \\ 0_{l \times k} & J_l - I_l \end{pmatrix}. \end{aligned} \quad (7.23)$$

Now, relaxation (7.13) simplifies to:

$$\begin{aligned} \text{maximize } & \frac{1}{2} \langle W, X_3 + X_4 \rangle \\ & X_1 + X_5 = I, \\ & \sum_{i=1}^6 X_i = J, \\ & \sum_{i=1}^6 m_i^{-1} A_i \otimes X_i \succeq 0, \\ & \langle J, X_i \rangle = m_i, X_{i^*} = X_i^\top, \text{ and } X_i \geq 0 \text{ for } i = 1, \dots, 6. \end{aligned} \quad (7.24)$$

The algebra spanned by  $A_1, \dots, A_6$  is isomorphic to the algebra  $\mathbb{C} \oplus \mathbb{C} \oplus \mathbb{C}^{2 \times 2}$ , so one may replace the matrices  $A_1, \dots, A_6$  above by their respective images under this isomorphism, which we denote by  $\phi$ . Their images are given by

$$\begin{aligned} \phi(A_1) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & \phi(A_2) &= \begin{pmatrix} -1 & 0 \\ 0 & k-1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & \phi(A_3) &= \sqrt{kl} \begin{pmatrix} 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \\ \phi(A_4) &= \sqrt{kl} \begin{pmatrix} 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, & \phi(A_5) &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}, & \phi(A_6) &= \begin{pmatrix} 0 & -1 \\ 0 & 0 \\ 0 & l-1 \end{pmatrix}. \end{aligned} \quad (7.25)$$

Feige and Langberg [6] and Han et al. [14] proposed another semidefinite programming relaxation for the maximum  $(k, l)$ -cut problem, namely

$$\begin{aligned} \text{maximize } & \frac{1}{4}\langle W, J - X \rangle \\ & X_{ii} = 1 \quad \text{for } i = 1, \dots, n, \\ & \langle J, X \rangle = (k - l)^2, \\ & X \succeq 0. \end{aligned} \tag{7.26}$$

Our relaxation (7.24) is actually at least as strong as (7.26). Indeed, if  $(X_1, \dots, X_6)$  is a feasible solution of (7.24), one may construct a feasible solution of (7.26) by setting

$$X = X_1 + X_2 - X_3 - X_4 + X_5 + X_6.$$

Indeed, it can be easily seen that  $X$  has diagonal entries equal to 1 and satisfies  $\langle J, X \rangle = (k - l)^2$ . To see that  $X$  is positive semidefinite, notice that the matrices  $A_1, \dots, A_6$  satisfy

$$A_1 + A_2 - A_3 - A_4 + A_5 + A_6 \succeq 0,$$

as one may easily check by using the isomorphism  $\phi$ . Then, from Theorem 7.10 one has that  $X$  is positive semidefinite.

Now, also  $\frac{1}{4}\langle W, J - X \rangle = \frac{1}{2}\langle W, X_3 + X_4 \rangle$ , so  $X$  has the same objective value as  $X_1, \dots, X_6$ , as we wanted.

Finally, our relaxation (7.24) is actually stronger than (7.26) in some cases. For example, consider the graph on five vertices with adjacency matrix:

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

For this example, a maximum  $(2, 3)$ -cut has weight 5, the Feige–Langberg upper bound (7.26) is 5.39099, and our relaxation (7.24) gives an upper bound of 5.00009.

To summarize, we have the following theorem.

**Theorem 7.12.** *The optimal value of (7.24) is at most that of (7.26), and can be strictly smaller for specific instances.*  $\square$

### 7.6.4 The Maximum Stable Set Problem

Let  $G = (V, E)$  be a graph with adjacency matrix  $A$  and stability number  $\alpha(G)$  and let an integer  $k > 1$  be given. One has  $\alpha(G) \geq k$ , if and only if

$$0 = \min \langle A_1 + A_2, P^T AP \rangle$$

$$P \in \Pi_n,$$

where  $A_1$  and  $A_2$  are from the coherent configuration in (7.23).

Applying our procedure we obtain the semidefinite programming problem:

$$\text{minimize } \langle A, X_1 + X_2 \rangle$$

$$X_1 + X_5 = I,$$

$$\sum_{i=1}^6 X_i = J,$$

$$\sum_{i=1}^6 m_i^{-1} \phi(A_i) \otimes X_i \geq 0,$$

$$\langle J, X_i \rangle = m_i, X_{i^*} = X_i^T, \text{ and } X_i \geq 0 \text{ for } i = 1, \dots, 6, \quad (7.27)$$

where the  $\phi(A_i)$  are as in (7.25). We denote by  $\sigma_k(G)$  the optimal value of (7.27) when  $A$  is the adjacency matrix of the graph  $G$ .

If the optimal value of this problem is strictly positive, then  $\alpha(G) < k$ . So an upper bound for  $\alpha(G)$  is given by

$$\max\{k : \sigma_k(G) = 0\}.$$

The bound provided above is at least as strong as the  $\vartheta'$  bound of McEliece et al. [23] and Schrijver [26]. Recall that, given a graph  $G = (V, E)$ , we let  $\vartheta'(G)$  be the optimal value of the semidefinite programming problem

$$\begin{aligned} & \text{maximize } \langle J, X \rangle \\ & \text{trace } X = 1, \\ & X_{uv} = 0 \quad \text{if } uv \in E, \\ & X \in \mathbb{R}^{V \times V} \text{ is nonnegative and positive semidefinite.} \end{aligned} \quad (7.28)$$

It is easy to see that

$$\max\{k : \sigma_k(G) = 0\} \leq \lfloor \vartheta'(G) \rfloor.$$

To see this, suppose  $\sigma_k(G) = 0$ . We show that  $\vartheta'(G) \geq k$ . So let  $X_1, \dots, X_6$  be an optimal solution of (7.27) for the parameter  $k$  and set  $X = k^{-1}(X_1 + X_2)$ . Then  $X$  is a feasible solution of (7.28).

Indeed,  $X$  is nonnegative, and it is also positive semidefinite since  $A_1 + A_2$  is a positive semidefinite matrix. Since the  $X_i$  sum up to  $J$  and  $X_1 + X_5 = I$ , we have that  $\text{trace } X = k^{-1} \text{trace } X_1 = k^{-1} \langle J, X_1 \rangle = 1$ . Finally since  $\langle A, X \rangle = k^{-1} \langle A, X_1 + X_2 \rangle = 0$  and  $X$  is nonnegative, we see that  $X_{uv} = 0$  whenever  $uv \in E$ , and therefore  $X$  is feasible for (7.28).

Now, notice that  $\langle J, X \rangle = k^{-1} \langle J, X_1 + X_2 \rangle = k$ , and we see that  $\vartheta'(G) \geq k$ , as we wanted.

### 7.6.5 The Vertex Separator Problem

Let  $G = (V, E)$  again be a graph with adjacency matrix  $A$  and  $|V| = n$ , and let  $n_1, n_2 > 0$  be given integers satisfying  $n_1 + n_2 < n$ .

The vertex separator problem is to find disjoint subsets  $V_1, V_2 \subset V$  such that  $|V_i| = n_i$  ( $i = 1, 2$ ) and no edge in  $E$  connects a vertex in  $V_1$  with a vertex in  $V_2$ , if such sets exist. (The set of vertices  $V \setminus (V_1 \cup V_2)$  is called a vertex separator in this case; see Feige et al. [7] for a review of approximation results for various separator problems.)

Such a vertex separator exists if and only if

$$\begin{aligned} 0 = \min \quad & \langle W, P^T AP \rangle \\ P \in \Pi_n, \end{aligned}$$

where

$$W = \begin{pmatrix} 0_{(n-n_1-n_2) \times (n-n_1-n_2)} & 0_{(n-n_1-n_2) \times n_1} & 0_{(n-n_1-n_2) \times n_2} \\ 0_{n_1 \times (n-n_1-n_2)} & 0_{n_1 \times n_1} & J_{n_1 \times n_2} \\ 0_{n_2 \times (n-n_1-n_2)} & J_{n_2 \times n_1} & 0_{n_2 \times n_2} \end{pmatrix},$$

and where the subscripts again indicate the matrix sizes. It is easy to verify that the matrix  $W$  belongs to a coherent configuration with  $d = 12$  relations, say  $\{A_1, \dots, A_{12}\}$ .

Thus one may again obtain a semidefinite programming problem where a strictly positive optimal value gives a certificate that the required vertex separator does not exist. The details of this example are left as an exercise to the reader.

## 7.7 Conclusion

In this chapter we have given a unified framework for deriving semidefinite programming relaxations of combinatorial problems. This approach is still in its infancy, and the relationships to various existing semidefinite programming relaxations is not yet fully understood. Moreover, since semidefinite programming relaxations play an important role in approximation algorithms, it would be very desirable to extend this framework with a general rounding technique that is amenable to analysis. It is the hope of the authors that these topics will spark the interest of other researchers in the optimization community.

**Acknowledgements** Etienne de Klerk would like to thank Chris Godsil for many valuable discussions on the contents of this chapter.

## References

1. Andersson, G.: An approximation algorithm for Max  $p$ -Section. Lecture Notes in Computer Science **1563**, 236–247 (1999)
2. Bannai, E., Ito, T.: Algebraic combinatorics. I: Association schemes. The Benjamin/Cummings Publishing Co. Inc., Menlo Park, CA (1984)
3. Brouwer, A.E., Cohen, A.M., Neumaier, A.: Distance-regular graphs. In: *Ergebnisse der Mathematik und ihrer Grenzgebiete (3) [Results in Mathematics and Related Areas (3)]*, vol. 18. Springer-Verlag, Berlin (1989)
4. Cvetković, D., Cangalović, M., Kovačević-Vujčić, V.: Semidefinite programming methods for the symmetric travelling salesman problem. In: *Proceedings of the 7th International Conference on Integer Programming and Combinatorial Optimization*, pp. 126–136. Springer-Verlag, London (1999)
5. Cameron, P.J.: Permutation Groups. London Mathematical Society Student Texts 45, Cambridge University Press, Cambridge (1999)
6. Feige U., Langberg, M.: Approximation algorithms for maximization problems arising in graph partitioning. *Journal of Algorithms* **41**, 174–211 (2001)
7. Feige, U., Lee, J.R., Hajiaghayi, M.T.: Improved approximation algorithms for minimum-weight vertex separators. *SIAM Journal on Computing* **38**(2), 629–657 (2008)
8. Frieze, A., Jerrum, M.: Improved approximation algorithms for MAX  $k$ -CUT and MAX BISECTION. *Algorithmica* **18**, 67–81 (1997)
9. The GAP Group. GAP – Groups, Algorithms, and Programming, Version 4.4.12. <http://www.gap-system.org> (2008)
10. Godsil, C.D.: Algebraic Combinatorics. Chapman & Hall, New York (1993)
11. Goemans, M.X., Williamson, D.P.: A general approximation technique for constrained forest problems. *SIAM Journal on Computing* **24**, 296–317 (1995)
12. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM* **42**, 1115–1145 (1995)
13. Greub, W.: Multilinear Algebra. Springer-Verlag, New York (1978)
14. Han, Q., Ye, Y., Zhang, J.: An improved rounding method and semidefinite relaxation for graph partitioning. *Mathematical Programming* **92**, 509–535 (2002)
15. Khot, S., Kindler, G., Mossel, E., O’Donnell, R.: Optimal inapproximability results for MAX-CUT and other 2-variable CSPs?. *SIAM Journal on Computing* **37**, 319–357 (2007)

16. de Klerk, E., Pasechnik, D.V., Sotirov, R.: On semidefinite programming relaxations of the travelling salesman problem. *SIAM Journal on Optimization* **19**, 1559–1573 (2008)
17. de Klerk, E., Sotirov, R.: Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. *Mathematical Programming* **122**(2), 225–246 (2010)
18. de Klerk, E., Sotirov, R.: Improved semidefinite programming bounds for quadratic assignment problems with suitable symmetry. *Mathematical Programming*, to appear.
19. Laurent, M., Rendl, F.: Semidefinite programming and integer programming. In: Aardal, K., Nemhauser, G., Weismantel, R. (eds.) *Handbook on Discrete Optimization*, pp. 393–514. Elsevier, Amsterdam (2005)
20. Lovász, L.: On the Shannon capacity of a graph. *IEEE Transactions on Information Theory* **25**, 1–7 (1979)
21. Manthey, B.: Minimum-weight cycle covers and their approximability. *Discrete Applied Mathematics* **157**, 1470–1480 (2009)
22. Martin, W.J., Tanaka, H.: Commutative association schemes. *European J. Combin.* **30**(6), 1497–1525 (2009)
23. McEliece, R.J., Rodemich, E.R., Rumsey, Jr., H.C.: The Lovász bound and some generalizations. *Journal of Combinatorics, Information & System Sciences*, 134–152 (1978)
24. Pasechnik, D.V., Kini, K.: A GAP package for computation with coherent configurations. In Fukuda, K., van der Hoeven, J., Joswig, M., Takayama, N. (eds.) *Mathematical Software - ICMS 2010, Third International Congress on Mathematical Software, Kobe, Japan, September 13–17, 2010. Proceedings*. Lecture Notes in Computer Science, vol. 6327, pp. 69–72. Springer (2010)
25. Povh J., Rendl, F.: Copositive and semidefinite relaxations of the quadratic assignment problem. *Discrete Optimization* **6**, 231–241 (2009)
26. Schrijver, A.: A comparison of the Delsarte and Lovász bounds. *IEEE Transactions on Information Theory* **25**, 425–429 (1979)
27. Schrijver, A.: New code upper bounds from the Terwilliger algebra. *IEEE Transactions on Information Theory* **51**, 2859–2866 (2005)
28. Wedderburn, J.H.M.: On hypercomplex numbers. *Proceedings of the London Mathematical Society* **6**, 77–118 (1907)
29. Ye, Y.: A .699 approximation algorithm for max-bisection. *Mathematical Programming* **90**, 101–111 (2001)
30. Zhao, Q., Karisch, S.E., Rendl, F., Wolkowicz, H.: Semidefinite programming relaxations for the quadratic assignment problem. *Journal of Combinatorial Optimization* **2**, 71–109 (1998)

# Chapter 8

## Copositive Programming

Samuel Burer

### 8.1 Introduction

A symmetric matrix  $S$  is *copositive* if  $y^T S y \geq 0$  for all  $y \geq 0$ , and the set of all copositive matrices, denoted  $C^*$ , is a closed, pointed, convex cone; see [25] for a recent survey. Researchers have realized how to model many NP-hard optimization problems as *copositive programs*, that is, programs over  $C^*$  for which the objective and all other constraints are linear [7, 9, 13, 16, 32–34]. This makes copositive programming NP-hard itself, but the models are nevertheless interesting because copositive programs are convex, unlike the problems which they model. In addition,  $C^*$  can be approximated up to any accuracy using a sequence of polyhedral-semidefinite cones of ever larger sizes [13, 30], so that an underlying NP-hard problem can be approximated up to any accuracy if one is willing to spend the computational effort.

In actuality, most of these NP-hard problems are modeled as linear programs over the dual cone  $C$  of *completely positive* matrices, that is, matrices  $Y$  that can be written as the sum of rank-1 matrices  $yy^T$  for  $y \geq 0$  [4]. These programs are called *completely positive programs*, and the aforementioned copositive programs are constructed using standard duality theory.

Currently the broadest class of problems known to be representable as completely positive programs are those with nonconvex quadratic objective and linear constraints over binary and continuous variables [9]. In addition, complementarity constraints on bounded, nonnegative variables can be incorporated. In this chapter, we recount and extend this result using the more general notion of matrices that are *completely positive over a closed, convex cone*.

---

S. Burer (✉)

Department of Management Sciences, University of Iowa, Iowa City, IA, 52245, USA  
e-mail: [samuel-burer@uiowa.edu](mailto:samuel-burer@uiowa.edu)

Recall the concept of a *linear conic program over  $\mathcal{K}$* , where  $\mathcal{K}$  is assumed to be a closed, convex cone [2]:

$$\min\{c^T x : Ax = b, x \in \mathcal{K}\}. \quad (8.1)$$

The standard form (8.1) is quite general. For example, it can be used to model mixed linear, second-order-cone, semidefinite programs with free variables. In such cases,  $\mathcal{K}$  is the Cartesian product of a nonnegative orthant, second-order cones, positive semidefinite cones, and a Euclidean space. The form of (8.1) is also critical for the design, analysis, and implementation of algorithms such as the simplex method when  $\mathcal{K}$  is a nonnegative orthant and interior-point methods more generally. For example, the iteration complexity of an interior-point method for (8.1) will depend on the self-concordancy barrier parameter for  $\mathcal{K}$  [29]. Loosely speaking, understanding  $\mathcal{K}$  is key for optimizing (8.1).

When faced with a new optimization problem, it thus seems prudent to determine if that problem can be modeled as a linear conic program over some (hopefully) well understood  $\mathcal{K}$ .

In this chapter, we show that any NP-hard *nonconvex quadratic conic program*

$$\nu_* := \min\{x^T Qx + 2c^T x : Ax = b, x \in \mathcal{K}\} \quad (8.2)$$

can be modeled as an explicit linear conic program over the closed, convex cone  $C$  of matrices that are completely positive over  $\mathbb{R}_+ \times \mathcal{K}$ , i.e., matrices  $Y$  that can be written as the sum of rank-1 matrices  $yy^T$  with  $y \in \mathbb{R}_+ \times \mathcal{K}$ .<sup>1</sup> We also extend this result to include certain types of nonconvex quadratic constraints. While  $C$  may be harder to understand than  $\mathcal{K}$  [3, 27], our approach provides a convex formulation for (8.2). For example, strong relaxations of  $C$  can be used to compute high quality lower bounds on  $\nu_*$ . In addition, (8.2) motivates the study of such cones  $C$ , particularly for common  $\mathcal{K}$  such as the Cartesian product of a nonnegative orthant, second-order cones, semidefinite cones, and a Euclidean space.<sup>2</sup>

The equivalence of (8.2) with a linear conic program over  $C$  is actually based on the characterization of a certain convex hull in  $C$ . This is the core of Sect. 8.2. We also extend in Sect. 8.2 the main convex hull result in several directions, e.g., the result is extended to incorporate certain types of nonconvex quadratic constraints in the variable  $x$ . Section 8.3 applies the convex hull results to (8.2) and related optimization problems and also presents some basic duality results. Section 8.4

<sup>1</sup>The paper [20] has established a similar result concurrently with this chapter, and the paper [19] studies the generalized notion of copositivity over an arbitrary set, analyzing important properties of the resulting convex cone.

<sup>2</sup>If  $\mathcal{K}$  is a semidefinite cone, than  $\mathcal{K}$  must be encoded in its “vectorized” form to match the development in this chapter. For example, the columns of a  $d \times d$  semidefinite matrix could be stacked into a single, long column vector of size  $d^2$ , and then the CP matrices  $yy^T$  over  $\mathbb{R}_+ \times \mathcal{K}$  would have size  $(d^2 + 1) \times (d^2 + 1)$ .

discusses techniques for working with  $C$ , particularly when  $\mathcal{K}$  is the nonnegative orthant  $\mathbb{R}_+^n$ , which has been the most studied case. Section 8.5 concludes the chapter with a few applications.

We make two important remarks. First, this chapter is not meant to be a complete survey of copositive programming. For this, please see the excellent, recent paper [17]. Instead, we intend only to give an introduction to some of the main results in the area, tainted with our own biases. Second, most research has examined the case when  $\mathcal{K} = \mathbb{R}_+^n$ . Our hope is that this chapter will stimulate further investigations for more general  $\mathcal{K}$ , especially when  $\mathcal{K}$  is the Cartesian product of a nonnegative orthant, second-order cones, semidefinite cones, and a Euclidean space.

## 8.2 Convex Hull Results

We first prove the main convex hull results that will be the basis of Sect. 8.3.

### 8.2.1 Main Result

Define the feasible set and recession cone of (8.2) to be

$$\mathcal{L} := \{x \in \mathcal{K} : Ax = b\}, \quad \mathcal{L}_\infty := \{d \in \mathcal{K} : Ad = 0\},$$

where  $A \in \mathbb{R}^{m \times n}$  and  $b \in \mathbb{R}^m$ . We assume  $\mathcal{L} \neq \emptyset$ , i.e., (8.2) is feasible. The set of *completely positive matrices* over the cone  $\mathbb{R}_+ \times \mathcal{K}$  is defined as

$$C := \left\{ \sum_k \begin{pmatrix} \zeta^k \\ z^k \end{pmatrix} \begin{pmatrix} \zeta^k \\ z^k \end{pmatrix}^T : \begin{pmatrix} \zeta^k \\ z^k \end{pmatrix} \in \mathbb{R}_+ \times \mathcal{K} \right\}.$$

$C$  is closed because  $\mathbb{R}_+ \times \mathcal{K}$  is closed [35, Lemma 1]. The representation of  $Y \in C$  in terms of  $(\zeta^k, z_k)$  is called a *completely positive decomposition*, and the decomposition is *proper* if  $(\zeta^k, z_k) \neq 0$  for all  $k$ . Also define the following two subsets of  $C$ :

$$\mathcal{L}^1 := \left\{ \begin{pmatrix} 1 \\ x \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix}^T : x \in \mathcal{L} \right\}, \quad \mathcal{L}_\infty^0 := \left\{ \begin{pmatrix} 0 \\ d \end{pmatrix} \begin{pmatrix} 0 \\ d \end{pmatrix}^T : d \in \mathcal{L}_\infty \right\}.$$

One can think of  $\mathcal{L}^1$  and  $\mathcal{L}_\infty^0$  as quadratic versions of  $\mathcal{L}$  and  $\mathcal{L}_\infty$ , respectively. The following proposition relates these two sets, where  $\text{conv}(\cdot)$  denotes the convex hull,  $\text{cone}(\cdot)$  denotes the convex conic hull, and  $\text{clconv}(\cdot)$  denotes the closure of the convex hull.

**Proposition 8.1.**  $\text{conv}(\mathcal{L}^1) + \text{cone}(\mathcal{L}_\infty^0) \subseteq \text{clconv}(\mathcal{L}^1)$ .

*Proof.* We represent an arbitrary element  $Y \in \text{conv}(\mathcal{L}^1) + \text{cone}(\mathcal{L}_\infty^0)$  via a finite sum

$$Y = \sum_{k \in P} \lambda_k \begin{pmatrix} 1 \\ x^k \end{pmatrix} \begin{pmatrix} 1 \\ x^k \end{pmatrix}^T + \sum_{k \in Z} \begin{pmatrix} 0 \\ d^k \end{pmatrix} \begin{pmatrix} 0 \\ d^k \end{pmatrix}^T$$

where  $P$  and  $Z$  are finite index sets,  $x^k \in \mathcal{L}$ ,  $d^k \in \mathcal{L}_\infty$ ,  $\lambda_k > 0$ , and  $\sum_{k \in P} \lambda_k = 1$ . Next let  $\bar{x} \in \mathcal{L}$  be fixed, and let  $\epsilon_k > 0$  for  $k \in P \cup Z$  be fixed such that  $\sum_{k \in P} \epsilon_k = \sum_{k \in Z} \epsilon_k$ . Define

$$Y^\epsilon := \sum_{k \in P} (\lambda_k - \epsilon_k) \begin{pmatrix} 1 \\ x^k \end{pmatrix} \begin{pmatrix} 1 \\ x^k \end{pmatrix}^T + \sum_{k \in Z} \epsilon_k \begin{pmatrix} 1 \\ \bar{x} + d^k / \sqrt{\epsilon_k} \end{pmatrix} \begin{pmatrix} 1 \\ \bar{x} + d^k / \sqrt{\epsilon_k} \end{pmatrix}^T.$$

If  $\epsilon$  is sufficiently close to the zero vector, then  $Y^\epsilon \in \text{conv}(\mathcal{L}^1)$ . Taking a sequence of such  $\epsilon$  converging to zero, we see  $Y^\epsilon \rightarrow Y$ , which implies  $Y \in \text{clconv}(\mathcal{L}^1)$ .  $\square$

We next introduce a third subset of  $C$ :

$$\mathcal{R} := \left\{ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \in C : \begin{array}{l} Ax = b \\ \text{diag}(AXA^T) = b \circ b \end{array} \right\}.$$

Here, the symbol  $\circ$  indicates the Hadamard product of vectors. Note that  $\mathcal{R}$  is closed and convex. By standard relaxation techniques,  $\mathcal{L}^1$  is contained in  $\mathcal{R}$ . So  $\text{clconv}(\mathcal{L}^1) \subseteq \mathcal{R}$ . We can prove more.

**Proposition 8.2.**  $\text{clconv}(\mathcal{L}^1) \subseteq \mathcal{R} \subseteq \text{conv}(\mathcal{L}^1) + \text{cone}(\mathcal{L}_\infty^0)$ .

*Proof.* The first containment holds by construction. To show the second, let

$$\begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} = \sum_k \begin{pmatrix} \zeta^k \\ z^k \end{pmatrix} \begin{pmatrix} \zeta^k \\ z^k \end{pmatrix}^T \quad (8.3)$$

be an arbitrary element of  $\mathcal{R}$  with proper completely positive decomposition. Partition the summands of (8.3) via the index sets  $P := \{k : \zeta^k > 0\}$  and  $Z := \{k : \zeta^k = 0\}$ . We claim: (i)  $k \in P$  implies  $z^k / \zeta^k \in \mathcal{L}$ ; (ii)  $k \in Z$  implies  $z^k \in \mathcal{L}_\infty$ .

To prove both parts of the claim, we need a technical result. From (8.3), we see

$$\sum_k (\zeta^k)^2 = 1. \quad (8.4)$$

Moreover, since  $Ax = b$  and  $\text{diag}(AXA^T) = b \circ b$ , we have

$$\begin{aligned} b &= \sum_k \zeta^k (Az^k) \\ b \circ b &= \sum_k \text{diag}(Az^k (z^k)^T A^T) = \sum_k (Az^k) \circ (Az^k). \end{aligned} \quad (8.5)$$

Thus

$$\left( \sum_k \zeta^k (Az^k) \right) \circ \left( \sum_k \zeta^k (Az^k) \right) = \left( \sum_k (\zeta^k)^2 \right) \left( \sum_k (Az^k) \circ (Az^k) \right)$$

and so by the equality-case of the Cauchy–Schwarz inequality, there exists  $\delta \in \Re^m$  such that, for all  $k$ ,

$$\zeta^k \delta = Az^k. \quad (8.6)$$

Claimed item (ii) follows directly from (8.6) and the fact that  $\zeta^k = 0$  when  $k \in Z$ . To prove (i), it suffices to show  $\delta = b$ . Indeed, (8.4), (8.5), and (8.6) imply

$$b = \sum_k \zeta^k (Az^k) = \sum_k \zeta^k (\zeta^k \delta) = \delta.$$

With claims (i) and (ii) established, we now complete the proof of the theorem. Taking  $\lambda_k := (\zeta^k)^2$ ,  $x^k := z^k / \zeta^k$  for all  $k \in P$ , and  $d^k := z^k$  for all  $k \in Z$ , we can write the completely positive decomposition (8.3) in the more convenient form

$$\begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} = \sum_{k \in P} \lambda_k \begin{pmatrix} 1 \\ x^k \end{pmatrix} \begin{pmatrix} 1 \\ x^k \end{pmatrix}^T + \sum_{k \in Z} \begin{pmatrix} 0 \\ d^k \end{pmatrix} \begin{pmatrix} 0 \\ d^k \end{pmatrix}^T \quad (8.7)$$

where  $\lambda_k > 0$ ,  $\sum_{k \in P} \lambda_k = 1$ ,  $x^k \in \mathcal{L}$ , and  $d^k \in \mathcal{L}_\infty$ .  $\square$

Propositions 8.1 and 8.2 combine to give the following key theorem.

**Theorem 8.1.**  $\mathcal{R} = \text{clconv}(\mathcal{L}^1)$ .

The proofs for Proposition 8.1–8.2 and Theorem 8.1 have been inspired by [5, 9].

### 8.2.2 Additional Implied Constraints

Because  $\mathcal{R}$  is contained in the positive semidefinite cone, the constraints  $Ax = b$  and  $\text{diag}(AXA^T) = b \circ b$  actually imply more. The following proposition and corollary were proved in [10], and very closely related results appear in [21].

**Proposition 8.3.** Suppose

$$Y = \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \geq 0$$

and define  $M := (b, -A)$  to be the matrix formed by concatenating  $b$  and  $-A$ . Then the following are equivalent:

- (i)  $Ax = b$ ,  $\text{diag}(AXA^T) = b \circ b$ .
- (ii)  $MYM^T = 0$ .
- (iii)  $MY = 0$ .

*Proof.* We will use the following equations:

$$\begin{aligned} MY &= \begin{pmatrix} b, -A \end{pmatrix} \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} = \begin{pmatrix} b - Ax & bx^T - AX \end{pmatrix} \\ MYM^T &= MY \begin{pmatrix} b^T \\ -A^T \end{pmatrix} = bb^T - Axb^T - bx^TA^T + AXA^T. \end{aligned} \quad (8.8)$$

(i)  $\Rightarrow$  (ii): We have  $MYM^T = AXA^T - bb^T$  with zero diagonal. Since  $Y$  is positive semidefinite, so is  $MYM^T$ . Hence,  $MYM^T = 0$ .

(ii)  $\Rightarrow$  (iii): Let  $Y = VV^T$  be a Gram representation of  $Y$ , which exists because  $Y \succeq 0$ . We have  $0 = \text{trace}(MYM^T) = \text{trace}(MVV^TM^T) = \|MV\|_F^2$ , where  $F$  indicates the Frobenius norm, and so  $MV = 0$ , which implies  $MY = (MV)V^T = 0$ .

(iii)  $\Rightarrow$  (i):  $Ax = b$  is clear from (8.8). Also  $AX = bx^T$ , which implies  $AXA^T = bb^T$ , so  $\text{diag}(AXA^T) = b \circ b$ .  $\square$

Because  $C$  is a subset of the positive semidefinite matrices, Proposition 8.3 implies that the constraints  $MYM^T = 0$  and  $MY = 0$  are redundant for  $\mathcal{R}$ .

**Corollary 8.1.** Define  $M := \begin{pmatrix} b, -A \end{pmatrix}$ . Then every  $Y \in \mathcal{R}$  satisfies the additional equations  $MYM^T = 0$  and  $MY = 0$ .

Proposition 8.3 also establishes that  $\mathcal{R}$  lacks interior, where by definition  $Y \in \mathcal{R}$  is interior if  $Y \in \text{int}(C)$ . Since  $C$  is contained in the positive semidefinite matrices,  $\text{int}(C)$  is contained in the positive definite matrices. As every  $Y \in \mathcal{R}$  has nontrivial null space as demonstrated by  $MY = 0$ , every  $Y \in \mathcal{R}$  is not positive definite, i.e.,  $\mathcal{R} \cap \text{int}(C) = \emptyset$ .

### 8.2.3 Extraneous Variables

It is sometimes possible to eliminate the variable  $x$  in  $\mathcal{R}$  and consequently express  $\mathcal{R}$  in terms of a slightly smaller cone  $C_0$  instead of  $C$ . The cone  $C_0$  is the set of matrices that are completely positive over  $\mathcal{K}$ :

$$C_0 := \left\{ \sum_k z^k (z^k)^T : z^k \in \mathcal{K} \right\}.$$

The key property we require is the following:

$$\exists y \in \Re^m \text{ s.t. } A^T y \in \mathcal{K}^*, \quad b^T y = 1, \quad (8.9)$$

where  $\mathcal{K}^* := \{s \in \Re^n : s^T x \geq 0 \forall x \in \mathcal{K}\}$  is the dual cone of  $\mathcal{K}$ . In this subsection, we assume (8.9) and define

$$\alpha := A^T y \in \mathcal{K}^*. \quad (8.10)$$

A direct consequence of (8.9) is that  $\alpha^T x = 1$  is redundant for  $\mathcal{L}$ . So Theorem 8.1 establishes that

$$\begin{aligned}\mathcal{R} &= \text{clconv}(\mathcal{L}^1) = \text{clconv}\left(\mathcal{L}^1 \cap \{x : \alpha^T x = 1\}\right) \\ &= \mathcal{R} \cap \left\{ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} : \begin{array}{l} \alpha^T x = 1 \\ \alpha^T X \alpha = 1 \end{array} \right\}.\end{aligned}$$

In other words,  $\alpha^T x = 1$  and  $\alpha^T X \alpha = 1$  are redundant for  $\mathcal{R}$ . Now applying Proposition 8.3, we see

$$(1, -\alpha^T) \begin{pmatrix} 1 \\ x \end{pmatrix} = 0 \iff x = X\alpha$$

and so

$$\mathcal{R} = \left\{ \begin{pmatrix} 1 & \alpha^T X \\ X\alpha & X \end{pmatrix} \in C : \begin{array}{l} AX\alpha = b \\ \text{diag}(AXA^T) = b \circ b \\ \alpha^T X \alpha = 1 \end{array} \right\}$$

Finally, the equations  $\alpha^T X \alpha = 1$  and

$$\begin{pmatrix} 1 & \alpha^T X \\ X\alpha & X \end{pmatrix} = (\alpha I)^T X (\alpha I)$$

along with  $\alpha \in \mathcal{K}^*$  demonstrate that

$$X \in C_0 \implies \begin{pmatrix} 1 & \alpha^T X \\ X\alpha & X \end{pmatrix} \in C.$$

Moreover, the converse holds because the bottom-right  $n \times n$  principal submatrix of a matrix in  $C$  is necessarily in  $C_0$ . Hence:

**Theorem 8.2.** *Suppose (8.9) holds, and define  $\alpha$  via (8.10). Then*

$$\mathcal{R} = \left\{ \begin{pmatrix} 1 & \alpha^T X \\ X\alpha & X \end{pmatrix} : X \in \mathcal{R}_0 \right\}$$

where

$$\mathcal{R}_0 := \left\{ X \in C_0 : \begin{array}{l} AX\alpha = b \\ \text{diag}(AXA^T) = b \circ b \\ \alpha^T X \alpha = 1 \end{array} \right\}.$$

In addition,  $\mathcal{R}_0 := \text{clconv}(\{xx^T : x \in \mathcal{L}\})$ .

Besides a more compact representation, an additional benefit of  $\mathcal{R}_0$  over  $\mathcal{R}$  is that  $\mathcal{R}_0$  may have an interior, whereas  $\mathcal{R}$  never does (see discussion in previous

subsection). This is an important feature of  $\mathcal{R}_0$  since the existence of an interior is generally a benefit both theoretically and computationally. However, it seems difficult to establish general conditions under which  $\mathcal{R}_0$  is guaranteed to have an interior. This is due in part to the general data  $(A, b)$  but also to the fact that not much is known about the structure of the interior of  $C_0$ . The paper [18] studies the case when  $\mathcal{K} = \mathbb{R}_+^n$ .

### 8.2.4 Quadratic Constraints

Consider a quadratic function  $x^T Fx + 2f^T x$ , and define

$$\phi_* := \min_{x \in \mathcal{L}} (x^T Fx + 2f^T x), \quad \phi^* := \max_{x \in \mathcal{L}} (x^T Fx + 2f^T x).$$

We wish to establish conditions under which a result similar to Theorem 8.1 holds for the further constrained feasible set

$$\mathcal{L}' := \mathcal{L} \cap \{x : x^T Fx + 2f^T x = \phi_*\}.$$

In addition to the sets  $\mathcal{L}_\infty$  and  $\mathcal{L}_\infty^0$  already defined in Sect. 8.2.1, define

$$\begin{aligned} (\mathcal{L}')^1 &:= \left\{ \begin{pmatrix} 1 \\ x \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix}^T : x \in \mathcal{L}' \right\} \\ \mathcal{R}' &:= \mathcal{R} \cap \left\{ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} : F \bullet X + 2f^T x = \phi_* \right\}, \end{aligned}$$

where  $\bullet$  indicates the trace inner product.

**Theorem 8.3.** *Suppose both  $\phi_*$  and  $\phi^*$  are finite and there exists  $\bar{x} \in \mathcal{L}'$  such that  $d^T(F\bar{x} + f) = 0$  for all  $d \in \mathcal{L}_\infty$ . Then  $\mathcal{R}' = \text{clconv}((\mathcal{L}')^1)$ .*

*Proof.* Analogous to Propositions 8.1 and 8.2, we argue

$$\begin{aligned} \text{conv}((\mathcal{L}')^1) + \text{cone}(\mathcal{L}_\infty^0) &\subseteq \text{clconv}((\mathcal{L}')^1) \\ \text{clconv}((\mathcal{L}')^1) &\subseteq \mathcal{R}' \subseteq \text{conv}((\mathcal{L}')^1) + \text{cone}(\mathcal{L}_\infty^0). \end{aligned}$$

To prove the first, we imitate the proof of Proposition 8.1, except here we specifically choose  $\bar{x}$  as hypothesized. The only thing to check is that  $Y^e \in \text{conv}((\mathcal{L}')^1)$  or, more specifically, that  $\bar{x} + d^k / \sqrt{\epsilon_k} \in \mathcal{L}'$ . It suffices to show  $\bar{x} + d \in \mathcal{L}'$  for all  $d \in \mathcal{L}_\infty$ . We already know  $\bar{x} + d \in \mathcal{L}$ . It remains to show

$$\begin{aligned} (\bar{x} + d)^T F(\bar{x} + d) + 2f^T(\bar{x} + d) &= \phi_* \iff \\ d^T Fd + 2d^T(F\bar{x} + f) &= 0 \iff \\ d^T Fd &= 0 \end{aligned}$$

which is true since  $-\infty < \phi_*$  and  $\phi^* < \infty$ ; otherwise,  $d$  would be a direction of recession to drive  $x^T Fx + 2f^T x$  to  $-\infty$  or  $\infty$ .

To prove the second, we imitate the proof of Proposition 8.2. The inclusion  $\text{conv}((\mathcal{L}')^1) \subseteq \mathcal{R}'$  is clear. For the second inclusion, the representation (8.7) holds without change. We next show that the constraint  $F \bullet X + 2f^T x = \phi_*$  implies  $(x^k)^T Fx^k + 2f^T x^k = \phi_*$  for all  $k \in P$ . From the previous paragraph, we know  $(d^k)^T Fd^k = 0$  for all  $k \in Z$ . Hence

$$\phi_* = F \bullet X + 2f^T x = \sum_{k \in P} \lambda_k ((x^k)^T Fx^k + 2f^T x^k).$$

By the definition of  $\phi_*$ , each summand on the right is at least  $\lambda_k \phi_*$ , and since  $\lambda_k > 0$  and  $\sum_{k \in P} \lambda_k = 1$ , it follows that each  $(x^k)^T Fx^k + 2f^T x^k = \phi_*$ , as desired.  $\square$

A common situation in which the condition  $d^T(F\bar{x} + f) = 0$  for all  $d \in \mathcal{L}_\infty$  occurs when  $\mathcal{L}$  is bounded and consequently  $\mathcal{L}_\infty = \{0\}$ . Another situation is when all variables  $x_j$  involved in  $x^T Fx + 2f^T x$  are bounded.

As an example, consider the quadratic equation  $x_i x_j = 0$  when  $\mathcal{L}$  implies both  $x_i$  and  $x_j$  are nonnegative and bounded. Then Theorem 8.3 shows that the constraint  $X_{ij} = 0$  in  $\mathcal{R}'$  captures the complementarity constraint  $x_i x_j = 0$ . Similarly, the binary condition  $x_i \in \{0, 1\}$  is captured by the equations  $x_i^2 = x_i$  and  $X_{ii} = x_i$  whenever  $\mathcal{L}$  implies  $0 \leq x_i \leq 1$ .

Theorem 8.3 also gives the following convex-hull result, which is significant because  $\mathcal{L}'$  is generally nonconvex.

**Corollary 8.2.**  $\text{clconv}(\mathcal{L}') = \left\{ x : \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \in \mathcal{R}' \text{ for some } X \right\}.$

Said differently, the closed convex hull of  $\mathcal{L}$  intersected with the equation  $x^T Fx + 2f^T x = \phi_*$  is the projection of  $\mathcal{R}'$  onto the coordinates corresponding to  $x$ .

Multiple quadratic constraints  $\{x^T F^j x + 2(f^j)^T x = (\phi^j)_*\}$ , where

$$(\phi^j)_* := \min\{x^T F^j x + 2(f^j)^T x : x \in \mathcal{L}\},$$

$$(\phi^j)^* := \max\{x^T F^j x + 2(f^j)^T x : x \in \mathcal{L}\},$$

may be easily incorporated as long as there exists  $\bar{x} \in \mathcal{L}$  satisfying all of the quadratic constraints and each quadratic constraint individually satisfies the assumptions of Theorem 8.3.

### 8.3 Optimization and Duality Results

In this section, we apply the convex hull results of Sect. 8.2 to the optimization (8.2) and related problems. We then discuss some basic duality results.

### 8.3.1 Optimization

By standard results, (8.2) may be expressed as

$$\nu_* = \min \left\{ \widehat{Q} \bullet Y : Y \in \text{clconv}(\mathcal{L}^1) \right\}, \quad \text{where } \widehat{Q} := \begin{pmatrix} 0 & c^T \\ c & Q \end{pmatrix}.$$

Thus, Theorem 8.1 implies

$$\nu_* = \min \left\{ \widehat{Q} \bullet Y : Y \in \mathcal{R} \right\}. \quad (8.11)$$

We formally state the relationship between (8.2) and (8.11) in the following corollary.

**Corollary 8.3.** *The nonconvex quadratic conic program (8.2) is equivalent to the linear conic program (8.11), i.e.: (i) both share the same optimal value; (ii) if*

$$Y^* = \begin{pmatrix} 1 & (x^*)^T \\ x^* & X^* \end{pmatrix}$$

*is optimal for (8.11), then  $x^*$  is in the convex hull of optimal solutions for (8.2).*

*Proof.* Item (i) follows from the preceding discussion. To prove (ii), we assume without loss of generality that  $\nu_* > -\infty$  and claim  $d^T Qd \geq 0$  for all  $d \in \mathcal{L}_\infty$ . If not, then there exists a nonnegative direction  $d$  along which the objective  $x^T Qx + 2c^T x$  can be driven to  $-\infty$ .

Item (ii) is then proved by examining the representation (8.7) for  $Y^*$ . In such a case, we must have  $x^k$  optimal for (8.2) for all  $k \in P$ ; otherwise,  $\widehat{Q} \bullet Y^*$  could not equal  $\nu_*$ . In fact, we know  $(z^k)^T Q z^k = 0$  for all  $k \in Z$ . Since  $x^* = \sum_{k \in P} \lambda_k x^k$ , item (ii) follows.  $\square$

Note that item (ii) of the corollary does not imply that  $x^*$  is itself optimal, just that it is a convex combination of optimal solutions.

In the context of Sect. 8.2.3, we also conclude that (8.2) is equivalent to

$$\min \left\{ Q \bullet X + 2c^T X \alpha : X \in \mathcal{R}_0 \right\} \quad (8.12)$$

as stated in the following corollary of Theorem 8.2.

**Corollary 8.4.** *Suppose (8.9) holds, and define  $\alpha$  via (8.10). Then (8.12) is equivalent to (8.2), i.e.: (i) both share the same optimal value; (ii) if  $X^*$  is optimal for (8.12), then  $X^* \alpha$  is in the convex hull of optimal solutions for (8.2).*

A similar results also holds for the situation of quadratic constraints discussed in Sect. 8.2.4 relative to the optimization problems

$$\min \left\{ x^T Qx + 2c^T x : \begin{array}{l} Ax = b, x \in \mathcal{K} \\ x^T Fx + 2f^T x = \phi_* \end{array} \right\} \quad (8.13)$$

$$\min \left\{ \widehat{Q} \bullet Y : Y \in \mathcal{R}' \right\}. \quad (8.14)$$

**Corollary 8.5.** Suppose both  $\phi_*$  and  $\phi^*$  are finite and there exists  $\bar{x} \in \mathcal{L}'$  such that  $d^T(F\bar{x} + f) = 0$  for all  $d \in \mathcal{L}_\infty$ . Then (8.14) is equivalent to (8.13), i.e.: (i) both share the same optimal value; (ii) if

$$Y^* = \begin{pmatrix} 1 & (x^*)^T \\ x^* & X^* \end{pmatrix}$$

is optimal for (8.14), then  $x^*$  is in the convex hull of optimal solutions for (8.13).

### 8.3.2 Duality

We now investigate some basic duality results for the linear conic problem (8.11), which is equivalent to (8.2) via Corollary 8.3. We first prove a technical detail that helps to interpret some of the results.

**Proposition 8.4.** Suppose  $A$  has full row rank. Then the constraints of problem (8.11) have the full-row-rank property.

*Proof.* The full-row-rank property for (8.11) is equivalent to linear independence of the following  $2m+1$  matrices ( $i = 1, \dots, m$ ):

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & a_i^T \\ a_i & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 0 & a_i a_i^T \end{pmatrix}$$

where  $a_i^T$  is the  $i$ -th row of  $A$ . Because the three types of matrices act in different portions of the matrix space, it suffices to show that the three types are separately independent. The first is a singleton; so independence is clear. For the second type, independence holds because  $A$  has full row rank. For the third, let  $d_i$  be multipliers such that

$$\sum_{i=1}^m d_i a_i a_i^T = 0 \iff A^T D A = 0$$

where  $D$  is the diagonal matrix containing  $d_i$ . Since  $A$  has full row rank,  $(AA^T)^{-1}$  exists, and so  $AA^T D A A^T = 0 \Leftrightarrow D = 0$ , as desired.  $\square$

The dual cone of  $C$  is

$$C^* := \{S : S \bullet Y \geq 0 \quad \forall Y \in C\}$$

$$= \left\{ S : \begin{pmatrix} \zeta \\ z \end{pmatrix}^T S \begin{pmatrix} \zeta \\ z \end{pmatrix} \geq 0 \quad \forall \begin{pmatrix} \zeta \\ z \end{pmatrix} \in \mathfrak{R}_+ \times \mathcal{K} \right\}$$

with interior

$$\text{int}(C^*) := \left\{ S : \begin{pmatrix} \zeta \\ z \end{pmatrix}^T S \begin{pmatrix} \zeta \\ z \end{pmatrix} > 0 \quad \forall 0 \neq \begin{pmatrix} \zeta \\ z \end{pmatrix} \in \mathfrak{R}_+ \times \mathcal{K} \right\}.$$

Note that  $\text{int}(C^*) \neq \emptyset$ ; for example, any positive definite  $S$  is an element of  $\text{int}(C^*)$ .

We use  $C^*$  to derive the dual of the formulation (8.11). By standard constructions, the dual is

$$\max \left\{ \lambda + b^T y + (b \circ b)^T w : \widehat{Q} - \begin{pmatrix} \lambda & \frac{1}{2} y^T A \\ \frac{1}{2} A^T y & A^T \text{Diag}(w) A \end{pmatrix} \in C^* \right\}. \quad (8.15)$$

In general, one needs to verify some constraint qualification in order to guarantee that strong duality holds. As mentioned in the previous subsection, (8.11) never has an interior. This implies that the level sets of (8.15) are unbounded under Proposition 8.4 [2], which includes the optimal solution set (if it exists). If (8.15) has interior, then the optimal value  $v_*$  of (8.11) is attained. One checkable, sufficient condition for (8.15) to have an interior is stated in the following proposition.

**Proposition 8.5.** *Suppose there exist  $\lambda, y, w$  such that*

$$\widehat{Q} - \begin{pmatrix} \lambda & \frac{1}{2} y^T A \\ \frac{1}{2} A^T y & A^T \text{Diag}(w) A \end{pmatrix} > 0.$$

*Then (8.15) has interior, and (8.11) attains its optimal value  $v_*$ .*

## 8.4 Working with the Cone $C$

In this section, we restrict our attention to cones  $\mathcal{K}$  that are the Cartesian product of a nonnegative orthant, second-order cones, semidefinite cones, and a Euclidean space.

### 8.4.1 Basic Results

In general, the cones  $C$  and  $C^*$  are intractable. For example, when  $\mathcal{K} = \mathfrak{R}_+^n$ , checking  $S \in C^*$  is co-NP complete [27]. On the other hand, some basic results are known or straightforward to prove. We define

$$\mathcal{I} := \{x \in \mathfrak{R}^n : x_1 \geq \|(x_2, \dots, x_n)\|\} \quad (\text{second-order or ‘ice cream’})$$

$$\mathcal{P} := \{X \text{ symmetric matrix} : X \succeq 0\} \quad (\text{positive semidefinite})$$

$$\mathcal{N} := \{X \text{ square matrix} : X \geq 0\} \quad (\text{nonnegative})$$

In each of the following propositions,  $C$  is the set of  $(n+1) \times (n+1)$  matrices, which are completely positive over  $\mathbb{R}_+ \times \mathcal{K}$ :

**Proposition 8.6.** *If  $\mathcal{K} = \mathbb{R}_+^n$ , then  $C \subseteq \mathcal{P} \cap \mathcal{N}$ . In addition, equality holds if and only if  $n \leq 3$ .*

*Proof.* The definition of  $X \in C$  implies  $X \in \mathcal{P} \cap \mathcal{N}$ . The low-dimension result is due to [26].  $\square$

To keep the dimensions clear, we caution the reader that  $\mathcal{P} \cap \mathcal{N}$  consists of size  $(n+1) \times (n+1)$  matrices.

**Proposition 8.7.** *If  $\mathcal{K} = \mathcal{I}^n$ , then*

$$C \subseteq \left\{ \begin{pmatrix} \chi & x^T \\ x & X \end{pmatrix} \succeq 0 : \begin{array}{l} x \in \mathcal{I}^n \\ X_{22} + \cdots + X_{nn} \leq X_{11} \end{array} \right\}.$$

*Proof.* Let  $\bar{x} \in \mathcal{I}$ . To prove the result, it suffices to show

$$\begin{pmatrix} \chi & x^T \\ x & X \end{pmatrix} := \begin{pmatrix} 1 \\ \bar{x} \end{pmatrix} \begin{pmatrix} 1 \\ \bar{x} \end{pmatrix}^T = \begin{pmatrix} 1 & \bar{x}^T \\ \bar{x} & \bar{x}\bar{x}^T \end{pmatrix}$$

is in the right-hand-side set. Positive semidefiniteness is clear, and so is  $x \in \mathcal{I}$ . The constraint  $X_{22} + \cdots + X_{nn} \leq X_{11}$  is equivalent to  $\bar{x}_2^2 + \cdots + \bar{x}_n^2 \leq \bar{x}_1^2$ , which is true because  $\bar{x} \in \mathcal{I}$ .  $\square$

In fact, [35] shows that

$$C_0 := \text{cone}(\{xx^T : x \in \mathcal{I}\}) = \{X \succeq 0 : X_{22} + \cdots + X_{nn} \leq X_{11}\}$$

which can be viewed as a strengthening of Proposition 8.7 in the bottom-right  $n \times n$  block.

**Proposition 8.8.** *If  $\mathcal{K} = \mathbb{R}^n$ , then  $C = \mathcal{P}$ .*

*Proof.*  $C \subseteq \mathcal{P}$  is clear. To prove the reverse inclusion, let  $Y \in \mathcal{P}$  and write a Gram representation  $Y = \sum_k y^k (y^k)^T$  for  $y^k \in \mathbb{R}^{n+1}$ . Without loss of generality, the first component of each  $y^k$  is nonnegative; if not, just negate  $y^k$  without affecting  $y^k (y^k)^T$ . This shows  $Y \in C$ .  $\square$

In the next subsection, we discuss further results for the case  $\mathcal{K} = \mathbb{R}_+^n$ . To our knowledge, however, the above are the only known results for  $\mathcal{K}$  involving Euclidean spaces and second-order cones. Nothing is known when  $\mathcal{K}$  is a semidefinite cone. In addition, the case when  $\mathcal{K}$  is the mixed Cartesian product of such cones has not been studied.

### 8.4.2 More When $\mathcal{K} = \mathbb{R}_+^n$

The case of the nonnegative orthant, i.e., when  $\mathcal{K} = \mathbb{R}_+^n$  and  $C$  is the cone of completely positive matrices, has received considerable attention in the literature. The recent monograph [4] studies  $C$  from the point of view of linear algebra, and the survey [25] covers  $C^*$  from the point of view of convex analysis. We focus on relatively recent results from the optimization point of view.

As mentioned in the Introduction, [13, 30] discuss a hierarchy of linear- and semidefinite-representable cones approximating  $C^*$  from the inside. More precisely, there exist closed, convex cones  $\mathcal{D}_r^*$  ( $r = 0, 1, 2, \dots$ ) such that  $\mathcal{D}_r^* \subset \mathcal{D}_{r+1}^*$  for all  $r \geq 0$  and  $\text{cl}(\cup_r \mathcal{D}_r^*) = C^*$ . The corresponding dual cones  $\mathcal{D}_r$  approximate  $C$  from the outside:  $\mathcal{D}_r \supset \mathcal{D}_{r+1}$  for all  $r$  and  $\cap_r \mathcal{D}_r = C$ . Explicit representations of the approximating cones have been worked out in [6, 30]. For example,  $\mathcal{D}_0$  is the cone of so-called *doubly nonnegative matrices* –  $\mathcal{P} \cap \mathcal{N}$  as introduced in Proposition 8.6. Moreover, using these approximating cones, [6, 22, 31] prove approximation results for several NP-hard problems. Variations of  $\mathcal{D}_r$  and  $\mathcal{D}_r^*$  have been presented in [31, 37], and adaptive polyhedral approximations similar in spirit have been presented in [8]. Another type of hierarchy is presented in [12].

A recent line of research has examined  $C$  for small values of  $n$ . This can shed light on larger completely positive matrices since principal submatrices are completely positive. According to Proposition 8.6,  $5 \times 5$  is the smallest size for which the doubly nonnegative matrices do *not* capture the completely positive matrices. (This corresponds to  $n = 4$  in our notation.) The papers [11, 15] provide closed-form inequalities to separate structured  $5 \times 5$  matrices in  $\mathcal{P} \cap \mathcal{N} \setminus C$ . [12] provides a separation algorithm for  $5 \times 5$  completely positive matrices, which establishes that  $5 \times 5$  completely positive matrices are tractable.

Computationally, approaches involving techniques of the two preceding paragraphs have mostly been limited to small or medium sized problems. For the approximating cones  $\mathcal{D}_r$  and  $\mathcal{D}_r^*$ , the size of the description of these cones grows exponentially with  $r$ , and even  $r = 0$  can present a challenge for off-the-shelf interior-point methods for, say,  $n \geq 100$ . Working with  $5 \times 5$  principal submatrices of an  $n \times n$  completely positive matrix also presents challenges because there are  $O(n^5)$  such submatrices.

As an alternative to interior-point methods, several large-scale algorithms [10, 36, 38] have been used to solve semidefinite programs over  $\mathcal{D}_0 = \mathcal{P} \cap \mathcal{N}$ . The key idea is to decouple the positive semidefinite constraint of  $\mathcal{P}$  from the nonnegativity constraint of  $\mathcal{N}$ , and then to devise an algorithmic framework that nevertheless handles the decoupling, e.g., a decomposition method. Successful results have been reported up to  $n \approx 1,000$ . Exploiting symmetry [14, 23] is also a promising technique to increase the size of solvable problems.

The authors of [8] also report the success of their adaptive algorithm for solving copositive programs on standard quadratic programs (see Sect. 8.5) up to size  $n = 10,000$ .

## 8.5 Applications

We close this chapter with a brief discussion of a few specific applications of linear conic programs over  $C$  that have appeared in the literature. The applications are given roughly in chronological order.

As far as we are aware, [7] establishes the first copositive representation of an NP-hard problem.

**Theorem 8.4.** *The standard quadratic program  $\min\{x^T Qx : e^T x = 1, x \geq 0\}$  is equivalent to the linear conic program  $\min\{Q \bullet X : e^T Xe = 1, X \in C_0\}$ , where  $C_0$  is the cone of matrices, which are completely positive over  $\mathcal{K} = \mathbb{R}_+^n$ , and  $e \in \mathbb{R}^n$  is the all-ones vector.*

The paper [13] shows that the NP-hard maximum stable set problem is a completely positive program.

**Theorem 8.5.** *Let  $G = (V, E)$  be an undirected graph with vertex set  $V = \{1, \dots, n\}$  and edge set  $E \subseteq V \times V$ . The maximum stable set problem on  $G$  is equivalent to the linear conic program*

$$\max \left\{ e^T X e : \begin{array}{l} X_{ij} = 0 \forall (i, j) \in E \\ \text{trace}(X) = 1, X \in C_0 \end{array} \right\}$$

where  $C_0$  is the cone of matrices, which are completely positive over  $\mathcal{K} = \mathbb{R}_+^n$ .

The authors also establish an explicit, finite bound on the size  $r$  which guarantees that the maximum stable set size is achieved (after rounding down) when  $\mathcal{D}_r$  is used as an approximation of the completely positive cone. Later papers [22, 31] improve upon this bound. Related to these results, [24] shows the following:

**Theorem 8.6.** *The chromatic number  $\chi$  of a graph  $G$  is the optimal value of a completely positive program.*

Related results can be found in [16].

In the thesis [32] and related papers [33, 34], it is shown that a certain class of quadratic programs over transportation matrices can be represented as completely positive programs. Transportation matrices are element-wise nonnegative with pre-specified row- and column-sums. One example of this is the quadratic assignment problem.

**Theorem 8.7.** *The quadratic assignment problem can be formulated as a completely positive program.*

We mention that the specific derivation of Theorem 8.7 in [34] is not subsumed by the techniques of this chapter (though the results herein can be used to give a second derivation of the theorem). However, Theorem 8.7 uses the Cauchy–Schwarz inequality just as the proof of Theorem 8.1 does.

The paper [9] contains the result on which this chapter is primarily based.

**Theorem 8.8.** Any nonconvex quadratic program having a mix of binary and continuous variables, as well as complementarity constraints on bounded variables, can be formulated as a completely positive program.

Using similar proof techniques, [28] shows the following:

**Theorem 8.9.** Consider a 0-1 integer program with uncertain objective vector, which nevertheless has known first moment vector and second moment matrix. The expected optimal value of the integer program can be formulated as a completely positive program.

This theorem finds applications in order statistics and project management.

Combining Theorem 8.1 and the low-dimensional case of Proposition 8.6, which establish  $C = \mathcal{P} \cap \mathcal{N}$  for  $n \leq 3$ , [1] investigates low-dimensional convex hulls.

**Theorem 8.10.** Let  $\mathcal{K} = \mathbb{R}_+^n$ . For  $n \leq 3$ ,

$$\text{clconv}(\mathcal{L}^1) = \left\{ \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix} \in \mathcal{P} \cap \mathcal{N} : \begin{array}{l} Ax = b \\ \text{diag}(AXA^T) = b \circ b \end{array} \right\}.$$

For  $n \leq 4$ , suppose in addition that (8.9) holds, and define  $\alpha$  via (8.10). Then

$$\text{clconv}(\mathcal{L}^1) = \left\{ \begin{pmatrix} 1 & \alpha^T X \\ X\alpha & X \end{pmatrix} : \begin{array}{l} AX\alpha = b \\ \text{diag}(AXA^T) = b \circ b \\ \alpha^T X\alpha = 1 \\ X \in \mathcal{P} \cap \mathcal{N} \end{array} \right\}.$$

The authors use this approach, for example, to derive a closed-form representation of

$$\text{clconv}\left(\left\{\binom{1}{x}\binom{1}{x}^T : 0 \leq x \leq e\right\}\right)$$

where  $x \in \mathbb{R}^2$ , which previously had only been partially characterized in the global optimization literature. To achieve the result, the authors add slack variables to form the system  $\{(x, s) \geq 0 : x + s = e\}$  and then apply the theorem with dimension  $n = 4$ .

**Acknowledgements** The author wishes to thank Mirjam Dür and Janez Povh for stimulating discussions on the topic of this chapter. The author also acknowledges the support of National Science Foundation Grant CCF-0545514.

## References

1. Anstreicher, K.M., Burer, S.: Computable representations for convex hulls of low-dimensional quadratic forms. Mathematical Programming (series B) **124**, 33–43 (2010)
2. Ben-Tal, A., Nemirovski, A.: Lectures on modern convex optimization: Analysis, algorithms, and engineering applications. MPS/SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA (2001)

3. Berman, A., Rothblum, U.G.: A note on the computation of the CP-rank. *Linear Algebra Appl.* **419**, 1–7 (2006)
4. Berman, A., Shaked-Monderer, N.: Completely Positive Matrices. World Scientific (2003)
5. Bomze, I., Jarre, F.: A note on Burer's copositive representation of mixed-binary QPs. *Optimization Letters* **4**, 465–472 (2010)
6. Bomze, I.M., de Klerk, E.: Solving standard quadratic optimization problems via linear, semidefinite and copositive programming. Dedicated to Professor Naum Z. Shor on his 65th birthday. *J. Global Optim.* **24**(2), 163–185 (2002)
7. Bomze, I.M., Dür, M., de Klerk, E., Roos, C., Quist, A.J., Terlaky, T.: On copositive programming and standard quadratic optimization problems. *J. Global Optim.* **18**(4), 301–320 (2000)
8. Bundfuss, S., Dür, M.: An adaptive linear approximation algorithm for copositive programs. *SIAM J. Optim.* **20**(1), 30–53 (2009)
9. Burer, S.: On the copositive representation of binary and continuous nonconvex quadratic programs. *Mathematical Programming* **120**, 479–495 (2009)
10. Burer, S.: Optimizing a polyhedral-semidefinite relaxation of completely positive programs. *Mathematical Programming Computation* **2**(1), 1–19 (2010)
11. Burer, S., Anstreicher, K.M., Dür, M.: The difference between  $5 \times 5$  doubly nonnegative and completely positive matrices. *Linear Algebra Appl.* **431**(9), 1539–1552 (2009)
12. Burer, S., Dong, H.: Separation and relaxation for cones of quadratic forms. Manuscript, University of Iowa (2010) Submitted to Mathematical Programming.
13. de Klerk, E., Pasechnik, D.V.: Approximation of the stability number of a graph via copositive programming. *SIAM J. Optim.* **12**(4), 875–892 (2002)
14. de Klerk, E., Sotirov, R.: Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. *Math. Programming* **122**(2, Ser. A), 225–246 (2010)
15. Dong, H., Anstreicher, K.: Separating Doubly Nonnegative and Completely Positive Matrices. Manuscript, University of Iowa (2010) To appear in Mathematical Programming. Available at [http://www.optimization-online.org/DB\\_HTML/2010/03/2562.html](http://www.optimization-online.org/DB_HTML/2010/03/2562.html).
16. Dukanovic, I., Rendl, F.: Copositive programming motivated bounds on the stability and the chromatic numbers. *Math. Program.* **121**(2, Ser. A), 249–268 (2010)
17. Dür, M.: Copositive Programming—A Survey. In: Diehl, M., Glineur, F., Jarlebring, E., Michalewski, W. (eds.) Recent Advances in Optimization and its Applications in Engineering, pp. 3–20. Springer (2010)
18. Dür, M., Still, G.: Interior points of the completely positive cone. *Electron. J. Linear Algebra* **17**, 48–53 (2008)
19. Eichfelder, G., Jahn, J.: Set-semidefinite optimization. *Journal of Convex Analysis* **15**, 767–801 (2008)
20. Eichfelder, G., Povh, J.: On reformulations of nonconvex quadratic programs over convex cones by set-semidefinite constraints. Manuscript, Faculty of Information Studies, Slovenia, December (2010)
21. Faye, A., Roupin, F.: Partial lagrangian relaxation for general quadratic programming. *4OR* **5**, 75–88 (2007)
22. Gvozdenović, N., Laurent, M.: Semidefinite bounds for the stability number of a graph via sums of squares of polynomials. In Lecture Notes in Computer Science, 3509, pp. 136–151. IPCO XI (2005)
23. Gvozdenović, N., Laurent, M.: Computing semidefinite programming lower bounds for the (fractional) chromatic number via block-diagonalization. *SIAM J. Optim.* **19**(2), 592–615 (2008)
24. Gvozdenović, N., Laurent, M.: The operator  $\Psi$  for the chromatic number of a graph. *SIAM J. Optim.* **19**(2), 572–591 (2008)
25. Hiriart-Urruty, J.-B., Seeger, A.: A variational approach to copositive matrices. *SIAM Review*, **52**(4), 593–629 (2010)
26. Maxfield, J.E., Minc, H.: On the matrix equation  $X'X = A$ . *Proc. Edinburgh Math. Soc. (2)* **13**, 125–129 (1962/1963)

27. Murty, K.G., Kabadi, S.N.: Some NP-complete problems in quadratic and nonlinear programming. *Math. Programming* **39**(2), 117–129 (1987)
28. Natarajan, K., Teo, C., Zheng, Z.: Mixed zero-one linear programs under objective uncertainty: A completely positive representation. *Operations Research*, **59**(3), 713–728, (2011)
29. Nesterov, Y.E., Nemirovskii, A.S.: *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, Philadelphia (1994)
30. Parrilo, P.: Structured Semidefinite Programs and Semi-algebraic Geometry Methods in Robustness and Optimization. PhD thesis, California Institute of Technology (2000)
31. Peña, J., Vera, J., Zuluaga, L.F.: Computing the stability number of a graph via linear and semidefinite programming. *SIAM J. Optim.* **18**(1), 87–105 (2007)
32. Povh, J.: Towards the optimum by semidefinite and copositive programming : new approach to approximate hard optimization problems. VDM Verlag (2009)
33. Povh, J., Rendl, F.: A copositive programming approach to graph partitioning. *SIAM J. Optim.* **18**(1), 223–241 (2007)
34. Povh, J., Rendl, F.: Copositive and semidefinite relaxations of the quadratic assignment problem. *Discrete Optim.* **6**(3), 231–241 (2009)
35. Sturm, J.F., Zhang, S.: On cones of nonnegative quadratic functions. *Math. Oper. Res.* **28**(2), 246–267 (2003)
36. Wen, Z., Goldfarb, D., Yin, W.: Alternating direction augmented lagrangian methods for semidefinite programming. Manuscript, Department of Industrial Engineering and Operations Research, Columbia University, New York, NY, USA (2009)
37. Yildirim, E.A.: On the accuracy of uniform polyhedral approximations of the copositive cone. Manuscript, Department of Industrial Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey (2009) To appear in Optimization Methods and Software.
38. Zhao, X., Sun, D., Toh, K.: A Newton-CG augmented Lagrangian method for semidefinite programming. *SIAM J. Optim.* **20**, 1737–1765 (2010)

# Chapter 9

## Invariant Semidefinite Programs

Christine Bachoc, Dion C. Gijswijt, Alexander Schrijver,  
and Frank Vallentin

### 9.1 Introduction

In the last years many results in the area of semidefinite programming were obtained for invariant semidefinite programs – semidefinite programs which have symmetries. This was done for a variety of problems and applications. The purpose of this handbook chapter is to give the reader the necessary background for dealing with semidefinite programs which have symmetry. Here the focus is on the basic theory and on representative examples. We do not aim at completeness of the presentation.

In all applications the underlying principles are similar: one simplifies the original semidefinite program which is invariant under a group action by applying an algebra isomorphism mapping a “large” matrix algebra to a “small” matrix algebra. Then it is sufficient to solve the semidefinite program using the smaller matrices.

---

C. Bachoc

Institut de Mathématiques de Bordeaux, Université Bordeaux I, 351, cours de la Libération,  
33405 Talence, France

e-mail: [bachoc@math.u-bordeaux1.fr](mailto:bachoc@math.u-bordeaux1.fr)

D.C. Gijswijt

CWI and Department of Mathematics, Leiden University; Centrum voor Wiskunde  
en Informatica (CWI), Sciencepark 123, 1098 XG Amsterdam, The Netherlands

e-mail: [dion.gijswijt@gmail.com](mailto:dion.gijswijt@gmail.com)

A. Schrijver

CWI and Department of Mathematics, University of Amsterdam; Centrum voor Wiskunde  
en Informatica (CWI), Sciencepark 123, 1098 XG Amsterdam, The Netherlands

e-mail: [lex@cwi.nl](mailto:lex@cwi.nl)

F. Vallentin (✉)

Delft Institute of Applied Mathematics, Technical University of Delft, 2600 GA Delft,  
The Netherlands

e-mail: [f.vallentin@tudelft.nl](mailto:f.vallentin@tudelft.nl)

We start this chapter by developing the general framework in the introduction where we give a step-by-step procedure for simplifying semidefinite programs: Especially Step 2 (first version), Step 2 (second version), and Step  $1\frac{1}{2}$  will be relevant in the later discussion. Both versions of Step 2 are based on the main structure theorem for matrix  $*$ -algebras. Step  $1\frac{1}{2}$  is based on the regular  $*$ -representation.

In Sect. 9.2 we give a proof of the main structure theorem for matrix  $*$ -algebras and we present the regular  $*$ -representation. Strictly speaking the framework of matrix  $*$ -algebras is slightly too general for the applications we have in mind. However, working with matrix  $*$ -algebras does not cause much extra work and it also gives a numerical algorithm for finding an explicit algebra isomorphism. Section 9.2 is mainly concerned with finite dimensional invariant semidefinite programs. In Sect. 9.3 we show how one can extend this to special classes of infinite dimensional invariant semidefinite programs, namely those which arise from permutation actions of compact groups. We focus on this case because of space limitations and because it suffices for our examples. This section is connected to Step 2 (second version) of the introduction.

The later sections contain examples coming from different areas: In Sect. 9.4 we consider finding upper bounds for finite error-correcting codes and in Sect. 9.5 we give lower bounds for the crossing number of complete bipartite graphs. Both applications are based on the methods explained in Sect. 9.2 (Step 2 (first version) and Step  $1\frac{1}{2}$  in the introduction). In Sect. 9.6 we use Step 2 (second version) for finding upper bounds for spherical codes and other geometric packing problems on the sphere. For this application the background in Sect. 9.3 is relevant. Section 9.4 and Sect. 9.6 both use the theta number of Lovász for finding upper bounds for the independence number of highly symmetric graphs. In Sect. 9.7 we show how one can exploit symmetry in polynomial optimization: We give particular sum of squares representations of polynomials which have symmetry.

This list of applications is not complete, and many more applications can be found in the literature. In the last Sect. 9.8 we give literature pointers to more applications.

### 9.1.1 Complex Semidefinite Programs

In order to present the theory as simple as possible we work with complex semidefinite programs. We give the necessary definitions. A complex matrix  $X \in \mathbb{C}^{n \times n}$  is *Hermitian* if  $X = X^*$ , where  $X^*$  is the *conjugate transpose* of  $X$ , i.e.  $X_{ij} = \overline{X_{ji}}$ . It is called *positive semidefinite*, we write  $X \succeq 0$ , if for all (column) vectors  $(\alpha_1, \dots, \alpha_n) \in \mathbb{C}^n$  we have

$$\sum_{i=1}^n \sum_{j=1}^n \alpha_i X_{ij} \overline{\alpha_j} \geq 0.$$

The space of complex matrices is equipped with a complex inner product, the trace product  $\langle X, Y \rangle = \text{trace}(Y^*X)$ , which is linear in the first entry. The inner product of two Hermitian matrices is always real.

**Definition 9.1.** A *(complex) semidefinite program* is an optimization problem of the form

$$\max\{\langle X, C \rangle : X \succeq 0, \langle X, A_1 \rangle = b_1, \dots, \langle X, A_m \rangle = b_m\}, \quad (9.1)$$

where  $A_1, \dots, A_m \in \mathbb{C}^{n \times n}$ , and  $C \in \mathbb{C}^{n \times n}$  are given Hermitian matrices,  $(b_1, \dots, b_m) \in \mathbb{R}^m$  is a given vector and  $X \in \mathbb{C}^{n \times n}$  is a variable Hermitian matrix.

A Hermitian matrix  $X \in \mathbb{C}^{n \times n}$  is called a *feasible solution* of (9.1) if it is positive semidefinite and fulfills all  $m$  linear constraints. It is called an *optimal solution* if it is feasible and if for every feasible solutions  $Y$  we have  $\langle X, C \rangle \geq \langle Y, C \rangle$ .

There is an easy reduction from complex semidefinite programs to semidefinite programs involving real matrices only, as noticed by Goemans and Williamson [42]. A complex matrix  $X \in \mathbb{C}^{n \times n}$  defines a real matrix

$$\begin{pmatrix} \Re(X) & -\Im(X) \\ \Im(X) & \Re(X) \end{pmatrix} \in \mathbb{R}^{2n \times 2n},$$

where  $\Re(X) \in \mathbb{R}^{n \times n}$  and  $\Im(X) \in \mathbb{R}^{n \times n}$  are the real and imaginary parts of  $X$ . Then the properties of being Hermitian and being complex positive semidefinite translate into being symmetric and being real positive semidefinite: We have for all real vectors  $\alpha = (\alpha_1, \dots, \alpha_{2n}) \in \mathbb{R}^{2n}$ :

$$\alpha^T \begin{pmatrix} \Re(X) & -\Im(X) \\ \Im(X) & \Re(X) \end{pmatrix} \alpha \geq 0.$$

On the other hand, complex semidefinite programs fit into the framework of conic programming (see e.g. Nemirovski [74]). Here one uses the cone of positive semidefinite Hermitian matrices instead of the cone of real positive semidefinite matrices. There are implementations available, SeDuMi (Sturm [87]) for instance, which can deal with complex semidefinite programs directly.

### 9.1.2 Semidefinite Programs Invariant Under a Group Action

Now we present the basic framework for simplifying a complex semidefinite program which has symmetry, i.e. which is invariant under the action of a group.

Let us fix some notation first. Let  $G$  be a finite group. Let  $\pi : G \rightarrow U_n(\mathbb{C})$  be a *unitary representation* of  $G$ , that is, a group homomorphism from the group  $G$  to the group of unitary matrices  $U_n(\mathbb{C})$ . The group  $G$  is acting on the set of Hermitian matrices by

$$(g, A) \mapsto \pi(g)A\pi(g)^*.$$

In general, a (left) *action* of a group  $G$  on a set  $M$  is a map

$$G \times M \rightarrow M, \quad (g, x) \mapsto gx,$$

that satisfies the following properties: We have  $1x = x$  for all  $x \in M$  where  $1$  denotes the neutral element of  $G$ . Furthermore,  $(g_1g_2)(x) = g_1(g_2x)$  for all  $g_1, g_2 \in G$  and all  $x \in M$ .

A matrix  $X$  is called  *$G$ -invariant* if  $X = gX$  for all  $g \in G$ , and we denote the set of all  $G$ -invariant matrices by  $(\mathbb{C}^{n \times n})^G$ . We say that the semidefinite program (9.1) is  *$G$ -invariant* if for every feasible solution  $X$  and for every  $g \in G$  the matrix  $gX$  is again a feasible solution and if it satisfies  $\langle gX, C \rangle = \langle X, C \rangle$  for all  $g \in G$ .

One example, which will receive special attention because of its importance, is the case of a permutation action: The set of feasible solutions is invariant under simultaneous permutations of rows and columns. Let  $G$  be a finite group which acts on the index set  $[n] = \{1, \dots, n\}$ . So we can see  $G$  as a subgroup of the permutation group on  $[n]$ . To a permutation  $\sigma$  on  $[n]$  corresponds a matrix permutation  $P_\sigma \in U_n(\mathbb{C})$  defined by

$$[P_\sigma]_{ij} = \begin{cases} 1, & \text{if } i = \sigma(j), \\ 0, & \text{otherwise.} \end{cases}$$

and the underlying unitary representation is  $\pi(\sigma) = P_\sigma$ . In this case, the action on matrices  $X \in \mathbb{C}^{n \times n}$  is

$$\sigma(X) = P_\sigma X P_\sigma^*, \text{ where } \sigma(X)_{ij} = X_{\sigma^{-1}(i), j} = X_{\sigma^{-1}(i), \sigma^{-1}(j)}.$$

In the following, we give some background on unitary representations. This part may be skipped at first reading. Let  $G$  be a finite group. A *representation* of  $G$  is a finite dimensional complex vector space  $V$  together with a homomorphism  $\pi : G \rightarrow \mathrm{GL}(V)$  from  $G$  to the group of invertible linear maps on  $V$ . The space  $V$  is also called a  *$G$ -space* or a  *$G$ -module* and  $\pi$  may be dropped in notations, replacing  $\pi(g)v$  by  $gv$ . In other words, the group  $G$  acts on  $V$  and this action has the additional property that  $g(\lambda v + \mu w) = \lambda gv + \mu gw$  for all  $(\lambda, \mu) \in \mathbb{C}^2$  and  $(v, w) \in V^2$ . The *dimension* of a representation is the dimension of the underlying vector space  $V$ .

If  $V$  is endowed with an inner product  $\langle v, w \rangle$  which is invariant under  $G$ , i.e. satisfies  $\langle gv, gw \rangle = \langle v, w \rangle$  for all  $(v, w) \in V^2$  and  $g \in G$ ,  $V$  is called a *unitary representation* of  $G$ . An inner product with this property always exists on  $V$ , since it is obtained by taking the average

$$\langle v, w \rangle = \frac{1}{|G|} \sum_{g \in G} \langle gv, gw \rangle_0$$

of an arbitrary inner product  $\langle v, w \rangle_0$  on  $V$ . So, with an appropriate choice of a basis of  $V$ , any representation of  $V$  is isomorphic to one of the form  $\pi : G \rightarrow U_n(\mathbb{C})$ , the form in which unitary representations of  $G$  were defined above.

### Step 1: Restriction to invariant subspace

Because of the convexity of (9.1), one can find an optimal solution of (9.1) in the set of  $G$ -invariant matrices. In fact, if  $X$  is an optimal solution of (9.1), so is its group average  $\frac{1}{|G|} \sum_{g \in G} gX$ . Hence, (9.1) is equivalent to

$$\max \left\{ \langle X, C \rangle : X \succeq 0, \langle X, A_1 \rangle = b_1, \dots, \langle X, A_m \rangle = b_m, X \in (\mathbb{C}^{n \times n})^G \right\}. \quad (9.2)$$

The  $G$ -invariant matrices intersected with the Hermitian matrices form a vector space. Let  $B_1, \dots, B_N$  be a basis of this space. Step 1 of simplifying a  $G$ -invariant semidefinite program is rewriting (9.2) in terms of this basis.

**Step 1:** If the semidefinite program (9.1) is  $G$ -invariant, then it is equivalent to

$$\begin{aligned} \max & \left\{ \langle X, C \rangle : x_1, \dots, x_N \in \mathbb{C}, \right. \\ & X = x_1 B_1 + \dots + x_N B_N \succeq 0, \\ & \left. \langle X, A_i \rangle = b_i, i = 1, \dots, m \right\}. \end{aligned} \quad (9.3)$$

In the case of a permutation action there is a canonical basis of  $(\mathbb{C}^{n \times n})^G$  which one can determine by looking at the orbits of the group action on pairs. Then, performing Step 1 using this basis amounts to coupling the variable matrix entries of  $X$ .

The orbit of the pair  $(i, j) \in [n] \times [n]$  under the group  $G$  is given by

$$O(i, j) = \{(\sigma(i), \sigma(j)) : \sigma \in G\}.$$

The set  $[n] \times [n]$  decomposes into the orbits  $R_1, \dots, R_M$  under the action of  $G$ . For every  $r \in \{1, \dots, M\}$  we define the matrix  $C_r \in \{0, 1\}^{n \times n}$  by  $(C_r)_{ij} = 1$  if  $(i, j) \in R_r$  and  $(C_r)_{ij} = 0$  otherwise. Then  $C_1, \dots, C_M$  forms a basis of  $(\mathbb{C}^{n \times n})^G$ , the *canonical basis*. If  $(i, j) \in R_r$  we also write  $C_{[i,j]}$  instead of  $C_r$ . Then,  $C_{[j,i]}^\top = C_{[i,j]}$ .

Note here that  $C_1, \dots, C_M$  is a basis of  $(\mathbb{C}^{n \times n})^G$ . In order to get a basis of the space of  $G$ -invariant Hermitian matrices we have to consider the orbits of unordered pairs: We get the basis by setting  $B_{\{i,j\}} = C_{[i,j]}$  if  $(i, j)$  and  $(j, i)$  are in the same orbit and  $B_{\{i,j\}} = C_{[i,j]} + C_{[j,i]}$  if they are in different orbits. The matrix entries of  $X$  in (9.3) are constant on the (unordered) orbits of pairs:  $X_{ij} = X_{\sigma(ij)} = X_{\sigma(ji)} = X_{ji}$  for all  $(i, j) \in [n] \times [n]$  and  $\sigma \in G$ .

### Step 2: Reducing the matrix sizes by block diagonalization

The  $G$ -invariant subspace  $(\mathbb{C}^{n \times n})^G$  is closed under matrix multiplication. This can be seen as follows: Let  $X, Y \in (\mathbb{C}^{n \times n})^G$  and let  $g \in G$ , then

$$g(XY) = \pi(g)XY\pi(g)^* = (\pi(g)X\pi(g)^*)(\pi(g)Y\pi(g)^*) = (gX)(gY) = XY.$$

Moreover, it is also closed under taking the conjugate transpose, since, for  $X \in (\mathbb{C}^{n \times n})^G$ ,  $\pi(g)X^*\pi(g)^* = (\pi(g)X\pi(g))^* = X^*$ . Thus  $(\mathbb{C}^{n \times n})^G$  has the structure of a *matrix  $*$ -algebra*. (However, not all matrix  $*$ -algebras are coming from group actions.)

In general, a *matrix  $*$ -algebra* is a set of complex matrices that is closed under addition, scalar multiplication, matrix multiplication, and taking the conjugate transpose. The main structure theorem of matrix  $*$ -algebras is the following:

**Theorem 9.1.** *Let  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  be a matrix  $*$ -algebra. There are numbers  $d$ , and  $m_1, \dots, m_d$  so that there is  $*$ -isomorphism between  $\mathcal{A}$  and a direct sum of complete matrix algebras*

$$\varphi : \mathcal{A} \rightarrow \bigoplus_{k=1}^d \mathbb{C}^{m_k \times m_k}.$$

We will give a proof of this theorem in Sect. 9.2.5 which also will give an algorithm for determining  $\varphi$ . In the case  $\mathcal{A} = (\mathbb{C}^{n \times n})^G$ , the numbers  $d$ , and  $m_1, \dots, m_d$  have a representation theoretic interpretation: The numbers are determined by the unitary representation  $\pi : G \rightarrow U_n(\mathbb{C})$ , where  $d$  is the number of pairwise non-isomorphic irreducible representations contained in  $\pi$  and where  $m_k$  is the multiplicity of the  $k$ -th isomorphism class of irreducible representations contained in  $\pi$ .

In the following we present background in representation theory which is needed for the understanding of the numbers  $d$  and  $m_1, \dots, m_d$  in Theorem 9.1. Again, this part may be skipped at first reading.

A *G-homomorphism*  $T$  between two  $G$ -spaces  $V$  and  $W$  is a linear map that commutes with the actions of  $G$  on  $V$  and  $W$ : for all  $g \in G$  and all  $v \in V$  we have  $T(gv) = gT(v)$ . If  $T$  is invertible, then it is a *G-isomorphism* and  $V$  and  $W$  are *G-isomorphic*. The set of all  $G$ -homomorphisms is a linear space, denoted  $\text{Hom}^G(V, W)$ . If  $V = W$ , then  $T$  is said to be a *G-endomorphism* and the set  $\text{End}^G(V)$  of  $G$ -endomorphisms of  $V$  is moreover an algebra under composition. If  $V = \mathbb{C}^n$ , and if  $G$  acts on  $\mathbb{C}^n$  by unitary matrices, then we have  $\text{End}^G(V) = (\mathbb{C}^{n \times n})^G = \mathcal{A}$ .

A representation  $\pi : G \rightarrow \text{Gl}(V)$  of  $G$  (and the corresponding  $G$ -space  $V$ ) is *irreducible* if it contains no proper subspace  $W$  such that  $gW \subset W$  for all  $g \in G$ , i.e.  $V$  contains no *G-subspace*. If  $V$  contains a proper  $G$ -subspace  $W$ , then also the orthogonal complement  $W^\perp$  relative to a  $G$ -invariant inner product, is a  $G$ -subspace and  $V$  is the direct sum  $W \oplus W^\perp$ . Inductively, one obtains Maschke's theorem:

*Every  $G$ -space is the direct sum of irreducible  $G$ -subspaces.*

This decomposition is generally not unique: For example, if  $G$  acts trivially on a vector space  $V$  (i.e.  $gv = v$  for all  $g \in G, v \in V$ ) of dimension at least 2, the irreducible subspaces are the 1-dimensional subspaces and  $V$  can be decomposed in many ways.

From now on, we fix a set  $\mathcal{R} = \{R_k : k = 1, \dots, d\}$  of representatives of the isomorphism classes of irreducible  $G$ -subspaces which are direct summands of  $V$ . Starting from an arbitrary decomposition of  $V$ , we consider, for  $k = 1, \dots, d$ , the sum of the irreducible subspaces which are isomorphic to  $R_k$ . One can prove that this  $G$ -subspace of  $V$ , is independent of the decomposition. It is called the *isotypic component* of  $V$  associated to  $R_k$  and is denoted  $\mathcal{MI}_k$ . The integer  $m_k$  such that  $\mathcal{MI}_k \simeq R_k^{m_k}$  is called the *multiplicity* of  $R_k$  in  $V$ . In other words, we have

$$V = \bigoplus_{k=1}^d \mathcal{MI}_k \quad \text{and} \quad \mathcal{MI}_k = \bigoplus_{i=1}^{m_k} H_{k,i}$$

where,  $H_{k,i}$  is isomorphic to  $R_k$ , and  $m_k \geq 1$ . The first decomposition is orthogonal with respect to an invariant inner product and is uniquely determined by  $V$ , while the decomposition of  $\mathcal{MI}_k$  is not unique unless  $m_k = 1$ .

*Schur's lemma* is the next crucial ingredient for the description of  $\text{End}^G(V)$ :

$$\text{If } V \text{ is } G\text{-irreducible, then } \text{End}^G(V) = \{\lambda \text{Id} : \lambda \in \mathbb{C}\} \simeq \mathbb{C}.$$

In the general case, when  $V$  is not necessarily  $G$ -irreducible,

$$\text{End}^G(V) \simeq \bigoplus_{k=1}^d \mathbb{C}^{m_k \times m_k}.$$

This result will be derived in the next section, in an algorithmic way, as a consequence of the more general theory of matrix  $*$ -algebras.

So we consider a  $*$ -isomorphism  $\varphi$  given by Theorem 9.1 applied to  $\mathcal{A} = (\mathbb{C}^{n \times n})^G$ :

$$\varphi : (\mathbb{C}^{n \times n})^G \rightarrow \bigoplus_{k=1}^d \mathbb{C}^{m_k \times m_k}. \quad (9.4)$$

Notice that since  $\varphi$  is a  $*$ -isomorphism between matrix algebras with unity,  $\varphi$  preserves also eigenvalues and hence positive semidefiniteness. Indeed, let  $X \in (\mathbb{C}^{n \times n})^G$  be  $G$ -invariant, then also  $X - \lambda I$  is  $G$ -invariant, and  $X - \lambda I$  has an inverse if and only if  $\varphi(X) - \lambda I$  has a inverse. This means that a test whether a (large)  $G$ -invariant matrix is positive semidefinite can be reduced to a test whether  $d$  (small) matrices are positive semidefinite. Hence, applying  $\varphi$  to (9.3) gives the second and final step of simplifying (9.1):

**Step 2 (first version):** *If the semidefinite program (9.1) is  $G$ -invariant, then it is equivalent to*

$$\begin{aligned} & \max \left\{ \langle X, C \rangle : x_1, \dots, x_N \in \mathbb{C}, \right. \\ & \quad X = x_1 B_1 + \dots + x_N B_N \succeq 0, \\ & \quad \langle X, A_i \rangle = b_i, \quad i = 1, \dots, m, \\ & \quad \left. x_1 \varphi(B_1) + \dots + x_N \varphi(B_N) \succeq 0 \right\}. \end{aligned} \quad (9.5)$$

Applying  $\varphi$  to a  $G$ -invariant semidefinite program is also called *block diagonalization*. The advantage of (9.5) is that instead of dealing with matrices of size  $n \times n$  one only has to deal with block diagonal matrices with  $d$  block matrices of size  $m_1, \dots, m_d$ , respectively. So one reduces the dimension from  $n^2$  to  $m_1^2 + \dots + m_d^2$ . In the case of a permutation action this sum is also the number of distinct orbits  $M$ . In many applications the latter is much smaller than the former.

In particular many practical solvers take advantage of the block structure to speed up the numerical calculations.

Instead of working with the  $*$ -isomorphism  $\varphi$  and a basis  $B_1, \dots, B_N$  of the Hermitian  $G$ -invariant matrices, one can also work with the inverse  $\varphi^{-1}$  and the standard basis of  $\bigoplus_{k=1}^d \mathbb{C}^{m_k \times m_k}$ . This is given by the matrices  $E_{k,uv} \in \mathbb{C}^{m_k \times m_k}$  where all entries of  $E_{k,uv}$  are zero except the  $(u,v)$ -entry which equals 1. This gives an *explicit parametrization* of the cone of  $G$ -invariant positive semidefinite matrices.

**Step 2 (second version):** *If the semidefinite program (9.1) is  $G$ -invariant, then it is equivalent to*

$$\max \left\{ \begin{aligned} & \langle X, C \rangle : X = \sum_{k=1}^d \sum_{u,v=1}^{m_k} x_{k,uv} \varphi^{-1}(E_{k,uv}) \\ & x_{k,uv} = \overline{x_{k,vu}}, \quad u, v = 1, \dots, m_k, \\ & (x_{k,uv})_{1 \leq u, v \leq m_k} \geq 0, \quad k = 1, \dots, d, \\ & \langle X, A_i \rangle = b_i, \quad i = 1, \dots, m \end{aligned} \right\}. \quad (9.6)$$

Hence, every  $G$ -invariant positive semidefinite matrix  $X$  is of the form

$$X = \sum_{k=1}^d \sum_{u,v=1}^{m_k} x_{k,uv} \varphi^{-1}(E_{k,uv}),$$

where the  $d$  matrices

$$X_k = (x_{k,uv})_{1 \leq u, v \leq m_k}, \quad k = 1, \dots, d,$$

are positive semidefinite. Define for  $(i, j) \in [n] \times [n]$  the matrix  $E_k(i, j) \in \mathbb{C}^{m_k \times m_k}$  componentwise by

$$[E_k(i, j)]_{uv} = [\varphi^{-1}(E_{k,uv})]_{ij}.$$

By definition we have  $E_k(i, j)^* = E_k(j, i)$ . Then, in the case of a permutation action,  $E_k(i, j) = E_k(\sigma(i), \sigma(j))$  for all  $(i, j) \in [n] \times [n]$  and  $\sigma \in G$ . With this notation one can write the  $(i, j)$ -entry of  $X$  by

$$X_{ij} = \sum_{k=1}^d \langle X_k, E_k(i, j) \rangle. \quad (9.7)$$

In summary, finding a block diagonalization of a  $G$ -invariant semidefinite program amounts to first identifying a basis of the Hermitian  $G$ -invariant matrices and then in finding an explicit  $*$ -isomorphism (9.4) between the algebra of

$G$ -invariant matrices and the direct sum of complete matrix algebras. In the following sections we will mainly be concerned with different strategies to find such a  $*$ -isomorphism.

### Step 1½: Reducing the matrix sizes by the regular $*$ -representation

In general finding a block diagonalization of a  $G$ -invariant semidefinite program is a non-trivial task, especially because one has to construct an explicit  $*$ -isomorphism. In cases where one does not have this one can fall back to a simpler  $*$ -isomorphism coming from the regular  $*$ -representation. In general this does not provide the maximum possible simplification. However, for instance in the case of a permutation action, it has the advantage that one can compute it on the level of knowing the orbit structure of the group action only.

For this we consider an orthogonal basis (with respect to the trace inner product  $\langle \cdot, \cdot \rangle$ ) of the  $G$ -invariant algebra  $(\mathbb{C}^{n \times n})^G$ . For instance, we can use the canonical basis  $C_1, \dots, C_M$  in the case of a permutation action. By considering the multiplication table of the algebra we define the *multiplication parameters*  $p_{rs}^t$ , sometimes also called *structural parameters*, by

$$C_r C_s = \sum_{t=1}^M p_{rs}^t C_t.$$

In the case of a permutation action the structural parameters can be computed by knowing the structure of orbits (if one chose the canonical basis):

$$p_{rs}^t = |\{k \in [n] : (i, k) \in R_r, (k, j) \in R_s\}|,$$

where  $(i, j) \in R_t$ . Here,  $p_{rs}^t$  does not depend on the choice of  $i$  and  $j$ . The norms  $\|C_r\| = \sqrt{\langle C_r, C_r \rangle}$  equal the sizes of the corresponding orbits. We define the matrices  $L(C_r)_{st} \in \mathbb{C}^{M \times M}$  by

$$(L(C_r))_{st} = \frac{\langle C_r C_t, C_s \rangle}{\|C_t\| \|C_s\|} = \frac{\|C_s\|}{\|C_t\|} p_{rt}^s.$$

**Theorem 9.2.** *Let  $\mathcal{L}$  the algebra generated by the matrices  $L(C_1), \dots, L(C_M)$ . Then the linear map*

$$\phi : (\mathbb{C}^{n \times n})^G \rightarrow \mathcal{L}, \quad \phi(C_r) = L(C_r), \quad r = 1, \dots, M,$$

*is a  $*$ -isomorphism.*

We will give a proof of this theorem in Sect. 9.2.6. There we will show that the  $*$ -isomorphism is the regular  $*$ -representation of the  $G$ -invariant algebra associated with the orthonormal basis  $C_1/\|C_1\|, \dots, C_M/\|C_M\|$ . Again, since  $\phi$  is a  $*$ -isomorphism between algebras with unity it preserves eigenvalues. This means that a test whether a  $G$ -invariant matrix of size  $n \times n$  is positive semidefinite can be reduced to testing whether an  $M \times M$  matrix is positive semidefinite.

**Step 1½ :** If the semidefinite program (9.1) is  $G$ -invariant, then it is equivalent to (9.5) where the  $*$ -isomorphism  $\varphi$  is replaced by  $\phi$ .

## 9.2 Matrix $*$ -Algebras

In Sect. 9.1.2 we saw that the process of block diagonalizing a semidefinite program can be naturally done in the framework of matrix  $*$ -algebras using the main structure theorem (Theorem 9.1). In this section we prove this main structure theorem. Although we are mainly interested in the case when the matrix  $*$ -algebra comes from a group, working in the more general framework here, does not cause much extra work. Furthermore the proof of the main structure theorem we give here provides an algorithmic way for finding a block diagonalization.

We start by giving the basic definitions, examples, and results of matrix  $*$ -algebras (Sects. 9.2.1–9.2.4). In Sect. 9.2.5 we prove the main structure theorem which gives a very efficient representation of a matrix  $*$ -algebra  $\mathcal{A}$ : We show that  $\mathcal{A}$  is  $*$ -isomorphic to a direct sum of full matrix  $*$ -algebras. The corresponding  $*$ -isomorphism is called a *block diagonalization* of  $\mathcal{A}$ . This corresponds to Step 2 in the introduction. After giving the proof we interpret it in the context of groups and we discuss a numerical algorithm for finding a block diagonalization which is based on the proof. In Sect. 9.2.6 we consider the regular  $*$ -representation, which embeds  $\mathcal{A}$  into  $\mathbb{C}^{M \times M}$ , where  $M = \dim \mathcal{A}$ . This corresponds to Step 1½ in the introduction.

### 9.2.1 Definitions and Examples

**Definition 9.2.** A *matrix  $*$ -algebra* is a linear subspace  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  of complex  $n \times n$  matrices, that is closed under matrix multiplication and under taking the conjugate transpose. The conjugate transpose of a matrix  $A$  is denoted  $A^*$ .

Matrix  $*$ -algebras are finite dimensional  $C^*$ -algebras and many results here can be extended to a more general setting. For a gentle introduction to  $C^*$ -algebras we refer to Takesaki [90].

Trivial examples of matrix  $*$ -algebras are the *full matrix algebra*  $\mathbb{C}^{n \times n}$  and the *zero algebra*  $\{0\}$ . From given matrix  $*$ -algebras  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  and  $\mathcal{B} \subseteq \mathbb{C}^{m \times m}$ , we can construct the *direct sum*  $\mathcal{A} \oplus \mathcal{B}$  and *tensor product*  $\mathcal{A} \otimes \mathcal{B}$  defined by

$$\mathcal{A} \oplus \mathcal{B} = \left\{ \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} : A \in \mathcal{A}, B \in \mathcal{B} \right\},$$

$$\mathcal{A} \otimes \mathcal{B} = \left\{ \sum_{i=1}^k A_i \otimes B_i : k \in \mathbb{N}, A_i \in \mathcal{A}, B_i \in \mathcal{B} \right\},$$

where  $A \otimes B \in \mathbb{C}^{nm \times nm}$  denotes the Kronecker- or tensor product. The *commutant* of  $\mathcal{A}$  is the matrix  $*$ -algebra

$$\mathcal{A}' = \{B \in \mathbb{C}^{n \times n} : BA = AB \text{ for all } A \in \mathcal{A}\}.$$

Many interesting examples of matrix  $*$ -algebras come from unitary group representations, as we already demonstrated in the introduction: Given a unitary representation  $\pi : G \rightarrow \mathrm{Gl}(n, \mathbb{C})$ , the set of invariant matrices  $(\mathbb{C}^{n \times n})^G = \{A \in \mathbb{C}^{n \times n} : \pi(g)A\pi(g)^{-1} = A\}$  is a matrix  $*$ -algebra. It is the commutant of the matrix  $*$ -algebra linearly spanned by the matrices  $\pi(g)$  with  $g \in G$ . If the unitary group representation is given by permutation matrices then the canonical basis of the algebra  $(\mathbb{C}^{n \times n})^G$  are the zero-one incidence matrices of orbits on pairs  $C_1, \dots, C_M$ , see Step 1 in Sect. 9.1.2.

Other examples of matrix  $*$ -algebras, potentially not coming from groups, include the (complex) Bose-Mesner algebra of an *association scheme*, see e.g. Bannai and Ito [12], and Brouwer et al. [19] and more generally, the adjacency algebra of a *coherent configuration*, see e.g. Cameron [21].

### 9.2.2 Commutative Matrix $*$ -Algebras

A matrix  $*$ -algebra  $\mathcal{A}$  is called *commutative* (or *Abelian*) if any pair of its elements commute:  $AB = BA$  for all  $A, B \in \mathcal{A}$ . Recall that a matrix  $A$  is *normal* if  $AA^* = A^*A$ . The spectral theorem for normal matrices states that if  $A$  is normal, there exists a unitary matrix  $U$  such that  $U^*AU$  is a diagonal matrix. More generally, a set of commuting normal matrices can be simultaneously diagonalized (see e.g. Horn and Johnson [47]). Since any algebra of diagonal matrices has a basis of zero-one diagonal matrices with disjoint support, we have the following theorem.

**Theorem 9.3.** *Let  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  be a commutative matrix  $*$ -algebra. Then there exist a unitary matrix  $U$  and a partition  $[n] = S_0 \cup S_1 \cup \dots \cup S_k$  with  $S_1, \dots, S_k$  nonempty, such that*

$$U^*\mathcal{A}U = \{\lambda_1 I_1 + \dots + \lambda_k I_k : \lambda_1, \dots, \lambda_k \in \mathbb{C}\},$$

where  $I_i$  is the zero-one diagonal matrix with ones in positions from  $S_i$  and zeroes elsewhere.

The matrices  $E_i = UI_iU^*$  satisfy  $E_0 + E_1 + \dots + E_k = I$ ,  $E_iE_j = \delta_{ij}E_i$  and  $E_i = E_i^*$ . The matrices  $E_1, \dots, E_k$  are the *minimal idempotents* of  $\mathcal{A}$  and form an orthogonal basis of  $\mathcal{A}$ . Unless  $S_0 = \emptyset$ ,  $E_0$  does not belong to  $\mathcal{A}$ .

Geometrically, we have an orthogonal decomposition

$$\mathbb{C}^n = V_0 \oplus V_1 \oplus \dots \oplus V_k,$$

where  $E_i$  is the orthogonal projection onto  $V_i$  or equivalently,  $V_i$  is the space spanned by the columns of  $U$  corresponding to  $S_i$ . The space  $V_0$  is the maximal subspace contained in the kernel of all matrices in  $\mathcal{A}$ .

### 9.2.3 Positive Semidefinite Elements

Recall that a matrix  $A$  is positive semidefinite ( $A \succeq 0$ ) if and only if  $A$  is *Hermitian* (that is  $A^* = A$ ) and all eigenvalues of  $A$  are nonnegative. Equivalently,  $A = U^*DU$  for some unitary matrix  $U$  and nonnegative diagonal matrix  $D$ .

Let  $\mathcal{A}$  be a matrix  $*$ -algebra. Positive semidefiniteness can be characterized in terms of  $\mathcal{A}$ .

**Proposition 9.1.** *An element  $A \in \mathcal{A}$  is positive semidefinite if and only if  $A = B^*B$  for some  $B \in \mathcal{A}$ .*

*Proof.* The “if” part is trivial. To see the “only if” part, let  $A \in \mathcal{A}$  be positive semidefinite and write  $A = U^*DU$  for some unitary matrix  $U$  and (nonnegative real) diagonal matrix  $D$ . Let  $p$  be a polynomial with  $p(\lambda_i) = \sqrt{\lambda_i}$  for all eigenvalues  $\lambda_i$  of  $A$ . Then taking  $B = p(A) \in \mathcal{A}$  we have  $B = U^*p(D)U$  and hence  $B^*B = U^*p(D)p(D)U = U^*DU = A$ .  $\square$

Considered as a cone in the space of Hermitian matrices in  $\mathcal{A}$ , the cone of positive semidefinite matrices is self-dual:

**Theorem 9.4.** *Let  $A \in \mathcal{A}$  be Hermitian. Then  $A \succeq 0$  if and only if  $\langle A, B \rangle \geq 0$  for all  $B \succeq 0$  in  $\mathcal{A}$ .*

*Proof.* Necessity is clear. For sufficiency, let  $A \in \mathcal{A}$  be Hermitian and let  $\mathcal{B} \subseteq \mathcal{A}$  be the  $*$ -subalgebra generated by  $A$ . By Theorem 9.3 we can write  $A = \lambda_1 E_1 + \dots + \lambda_k E_k$ , where the  $E_i$  are the minimal idempotents of  $\mathcal{B}$ . The  $E_i$  are positive semidefinite since their eigenvalues are zero or one. Hence by assumption,  $\lambda_i = \frac{\langle A, E_i \rangle}{\langle E_i, E_i \rangle} \geq 0$ . Therefore  $A$  is a nonnegative linear combination of positive semidefinite matrices and hence positive semidefinite.  $\square$

As a corollary we have:

**Corollary 9.1.** *The orthogonal projection  $\pi_{\mathcal{A}} : \mathbb{C}^{n \times n} \rightarrow \mathcal{A}$  preserves positive semidefiniteness.*

This implies that if the input matrices of the semidefinite program (9.1) lie in some matrix  $*$ -algebra, then we can assume that the optimization variable  $X$  lies in the same matrix  $*$ -algebra: If (9.1) is given by matrices  $C, A_1, \dots, A_m \in \mathcal{A}$  for some matrix  $*$ -algebra  $\mathcal{A}$ , the variable  $X$  may be restricted to  $\mathcal{A}$  without changing the objective value. Indeed, any feasible  $X$  can be replaced by  $\pi_{\mathcal{A}}(X)$ , which is again feasible and has the same objective value. When  $\mathcal{A}$  is the invariant algebra of a group, this amounts to replacing  $X$  by the average under the action of the group.

### 9.2.4 *\*-Homomorphisms and Block Diagonalization*

**Definition 9.3.** A map  $\phi : \mathcal{A} \rightarrow \mathcal{B}$  between two matrix  $*$ -algebras  $\mathcal{A}$  and  $\mathcal{B}$  is called a  $*$ -homomorphism if

- (i)  $\phi$  is linear,
- (ii)  $\phi(AB) = \phi(A)\phi(B)$  for all  $A, B \in \mathcal{A}$ ,
- (iii)  $\phi(A^*) = \phi(A)^*$  for all  $A \in \mathcal{A}$ .

If  $\phi$  is a bijection, the inverse map is also a  $*$ -homomorphism and  $\phi$  is called a  $*$ -isomorphism.

It follows directly from Proposition 9.1 that  $*$ -homomorphisms preserve positive semidefiniteness: Let  $\phi : \mathcal{A} \rightarrow \mathcal{B}$  be a  $*$ -homomorphism. Then  $\phi(A)$  is positive semidefinite for every positive semidefinite  $A \in \mathcal{A}$ .

The implication of this for semidefinite programming is the following. Given a semidefinite program with matrix variable restricted to a matrix  $*$ -algebra  $\mathcal{A}$  and a  $*$ -isomorphism  $\mathcal{A} \rightarrow \mathcal{B}$ , we can rewrite the semidefinite program in terms of matrices in  $\mathcal{B}$ . This can be very useful if the matrices in  $\mathcal{B}$  have small size compared to those in  $\mathcal{A}$ . In the following, we will discuss two such efficient representations of a general matrix  $*$ -algebra.

### 9.2.5 *Block Diagonalization*

In this section, we study the structure of matrix  $*$ -algebras in some more detail. The main result is, that any matrix  $*$ -algebra is  $*$ -isomorphic to a direct sum of full matrix algebras:

**Theorem 9.5 (= Theorem 9.1).** *Let  $\mathcal{A} \subseteq \mathbb{C}^{n \times n}$  be a matrix  $*$ -algebra. There are numbers  $d$ , and  $m_1, \dots, m_d$  so that there is  $*$ -isomorphism between  $\mathcal{A}$  and a direct sum of complete matrix algebras*

$$\varphi : \mathcal{A} \rightarrow \bigoplus_{k=1}^d \mathbb{C}^{m_k \times m_k}.$$

This theorem is well-known in the theory of  $C^*$ -algebras, where it is generalized to  $C^*$ -algebras of compact operators on a Hilbert space (cf. Davidson [27, Chap. I.10]).

Some terminology: Let  $\mathcal{A}$  be a matrix  $*$ -algebra. A matrix  $*$ -algebra  $\mathcal{B}$  contained in  $\mathcal{A}$  is called a  $*$ -subalgebra of  $\mathcal{A}$ . An important example is the center of  $\mathcal{A}$  defined by

$$C_{\mathcal{A}} = \{A \in \mathcal{A} : AB = BA \text{ for all } B \in \mathcal{A}\}.$$

There is a unique element  $E \in \mathcal{A}$  such that  $EA = AE = A$  for every  $A \in \mathcal{A}$ , which is called the *unit* of  $\mathcal{A}$ . If  $\mathcal{A}$  is non-zero and  $C_{\mathcal{A}} = \mathbb{C}E$  (equivalently:  $\mathcal{A}$  has no nontrivial *ideal*), the matrix  $*$ -algebra  $\mathcal{A}$  is called *simple*.

We shall prove that every matrix  $*$ -algebra is the direct sum of simple matrix  $*$ -algebras:

$$\mathcal{A} = \bigoplus_{i=1}^d E_i \mathcal{A},$$

where the  $E_i$  are the minimal idempotents of  $C_{\mathcal{A}}$ . Every simple matrix  $*$ -algebra is  $*$ -isomorphic to a full matrix algebra. Together these facts imply Theorem 9.5.

To conclude this section, we will give an elementary and detailed proof of Theorem 9.5.

*Proof.* Let  $\mathcal{B} \subseteq \mathcal{A}$  be an inclusionwise maximal commutative  $*$ -subalgebra of  $\mathcal{A}$ . Then any  $A \in \mathcal{A}$  that commutes with every element of  $\mathcal{B}$ , is itself an element of  $\mathcal{B}$ . Indeed, if  $A$  is normal, this follows from the maximality of  $\mathcal{B}$  since  $\mathcal{B} \cup \{A, A^*\}$  generates a commutative  $*$ -algebra containing  $\mathcal{B}$ . If  $A$  is not normal, then  $A + A^* \in \mathcal{B}$  by the previous argument and hence  $A(A + A^*) = (A + A^*)A$ , contradicting the fact that  $A$  is not normal.

By replacing  $\mathcal{A}$  by  $U^* \mathcal{A} U$  for a suitable unitary matrix  $U$ , we may assume that  $\mathcal{B}$  is in diagonal form as in Theorem 9.3. For any  $A \in \mathcal{A}$  and  $i, j = 0, \dots, k$ , denote by  $A_{ij} \in \mathbb{C}^{|S_i| \times |S_j|}$  the restriction of  $I_i A I_j$  to the rows in  $S_i$  and columns in  $S_j$ . Let  $A \in \mathcal{A}$  and  $i, j \in \{0, \dots, k\}$ . We make the following observations:

- (i)  $A_{ii}$  is a multiple of the identity matrix and  $A_{00} = 0$ ,
- (ii)  $A_{0i}$  and  $A_{i0}$  are zero matrices,
- (iii)  $A_{ij}$  is either zero or a nonzero multiple of a (square) unitary matrix.

Item (i) follows since  $I_i A I_i$  commutes with  $I_0, \dots, I_k$  and therefore belongs to  $\mathcal{B}$ . Hence  $I_i A I_i$  is a multiple of  $I_i$  and  $I_0 A I_0 = 0$  since  $I_0 \mathcal{B} I_0 = \{0\}$ . Similarly,  $I_0 A A^* I_0 = 0$ , which implies that  $I_0 A = 0$ , showing (ii). For item (iii), suppose that  $A_{ij}$  is nonzero and assume without loss of generality that  $|S_i| \geq |S_j|$ . Then by (i),  $A_{ij} A_{ij}^* = \lambda I$  for some positive real  $\lambda$ , and therefore has rank  $|S_i|$ . This implies that  $|S_j| = |S_i|$  and  $\sqrt{\lambda} \cdot A_{ij}$  is unitary.

Observe that (ii) shows that  $I_1 + \dots + I_k$  is the unit of  $\mathcal{A}$ .

Define the relation  $\sim$  on  $\{1, \dots, k\}$  by setting  $i \sim j$  if and only if  $I_i \mathcal{A} I_j \neq \{0\}$ . This is an equivalence relation. Indeed,  $\sim$  is reflexive by (i) and symmetric since  $I_i \mathcal{A} I_j = (I_j \mathcal{A} I_i)^*$ . Transitivity follows from (iii) since  $I_h \mathcal{A} I_j \supseteq (I_h \mathcal{A} I_i)(I_i \mathcal{A} I_j)$  and the product of two unitary matrices is unitary.

Denote by  $\{E_1, \dots, E_d\} = \{\sum_{j \sim i} I_j : i = 1, \dots, k\}$  the zero-one diagonal matrices induced by the equivalence relation. Since the center  $C_{\mathcal{A}}$  of  $\mathcal{A}$  is contained in  $\mathcal{B} = \mathbb{C}I_1 + \dots + \mathbb{C}I_k$ , it follows by construction that  $E_1, \dots, E_d$  span the center, and are its minimal idempotents. We find the following block structure of  $\mathcal{A}$ :

$$\mathcal{A} = \{0\} \oplus E_1 \mathcal{A} \oplus \dots \oplus E_d \mathcal{A},$$

where the matrix  $*$ -algebras  $E_i\mathcal{A}$  are simple. For the rest of the proof we may assume that  $\mathcal{A}$  is simple ( $d = 1$ ) and that  $E_0 = 0$ .

Since  $\sim$  has only one equivalence class, for every matrix  $A = (A_{ij})_{i,j=1}^k \in \mathcal{A}$ , the “blocks”  $A_{ij}$  are square matrices of the same size. Furthermore, we can fix an  $A \in \mathcal{A}$  for which all the  $A_{ij}$  are unitary. For any  $B \in \mathcal{A}$ , we have  $A_{1i}B_{ij}(A_{1j})^* = (AI_iBI_jA^*)_{11}$ . By (i), it follows that  $\{A_{1i}B_{ij}(A_{1j})^* : B \in \mathcal{B}\} = \mathbb{C}I$ . Hence setting  $U$  to be the unitary matrix  $U := \text{diag}(A_{11}, \dots, A_{1k})$ , we see that  $U\mathcal{A}U^* = \{(a_{ij}I)_{i,j=1}^k : a_{i,j} \in \mathbb{C}\}$ , which shows that  $\mathcal{A}$  is  $*$ -isomorphic to  $\mathbb{C}^{k \times k}$ .  $\square$

### 9.2.5.1 Relation to Group Representations

In the case that  $\mathcal{A} = (\mathbb{C}^{n \times n})^G$ , where  $\pi : G \rightarrow U_n(\mathbb{C})$  is a unitary representation, the block diagonalization can be interpreted as follows. The diagonalization of the maximal commutative  $*$ -subalgebra  $\mathcal{B}$ , gives a decomposition  $\mathbb{C}^n = V_1 \oplus \dots \oplus V_k$  into irreducible submodules. Observe that  $V_0 = \{0\}$  since  $\mathcal{A}$  contains the identity matrix. The equivalence relation  $\sim$  yields the isotypic components  $\text{Im}E_1, \dots, \text{Im}E_d$ , where the sizes of the equivalence classes correspond to the block sizes  $m_i$  in Theorem 9.5. Here Schur’s lemma is reflected by the fact that  $I_i\mathcal{A}I_j = \mathbb{C}I_i$  if  $i \sim j$  and  $\{0\}$  otherwise.

### 9.2.5.2 Algorithmic Aspects

If a matrix  $*$ -algebra  $\mathcal{A}$  is given explicitly by a basis, then the above proof of Theorem 9.5 can be used to find a block diagonalization of  $\mathcal{A}$  computationally. Indeed, it suffices to find an inclusion-wise maximal commutative  $*$ -subalgebra  $\mathcal{B} \subseteq \mathcal{A}$  and compute a common system of eigenvectors for (basis) elements of  $\mathcal{B}$ . This can be done by standard linear algebra methods. For example finding a maximal commutative  $*$ -subalgebra of  $\mathcal{A}$  can be done by starting with  $\mathcal{B} = \langle A \rangle$ , the  $*$ -subalgebra generated by an arbitrary Hermitian element in  $\mathcal{A}$ . As long as  $\mathcal{B}$  is not maximal, there is a Hermitian element in  $\mathcal{A} \setminus \mathcal{B}$  that commutes with all elements in (a basis of)  $\mathcal{B}$ , hence extending  $\mathcal{B}$ . Such an element can be found by solving a linear system in  $O(\dim \mathcal{A})$  variables and  $O(\dim \mathcal{B})$  constraints. In at most  $\dim \mathcal{A}$  iterations, a maximal commutative  $*$ -subalgebra is found.

Practically more efficient is to find a “generic” Hermitian element  $A \in \mathcal{A}$ . Then the matrix  $*$ -algebra  $\mathcal{B}$  generated by  $A$  will be a maximal commutative  $*$ -subalgebra of  $\mathcal{A}$  and diagonalizing the matrix  $A$  also diagonalizes  $\mathcal{B}$ . Such a generic element can be found by taking a random Hermitian element from  $\mathcal{A}$  (with respect to the basis), see Murota et. al. [72]. If a basis for the center of  $\mathcal{A}$  is known a priori (or by solving a linear system in  $O(\dim \mathcal{A})$  variables and equations), as an intermediate step the center could be diagonalized, followed by a block diagonalization of the simple components of  $\mathcal{A}$ , see Dobre et al. [33].

### 9.2.6 Regular \*-Representation

Let  $\mathcal{A}$  be a matrix  $*$ -algebra of dimension  $M$  and let  $C_1, \dots, C_M$  be an orthonormal basis of  $\mathcal{A}$  with respect to the trace product  $\langle \cdot, \cdot \rangle$ . For fixed  $A \in \mathcal{A}$ , left-multiplication by  $A$  defines a linear map  $B \mapsto AB$  on  $\mathcal{A}$ . With respect to the orthonormal basis  $C_1, \dots, C_M$ , this linear map is represented by the matrix  $L(A) \in \mathbb{C}^{M \times M}$  given by

$$L(A)_{st} = \langle AC_t, C_s \rangle.$$

The map

$$L : \mathcal{A} \rightarrow \mathbb{C}^{M \times M}$$

is an injective  $*$ -homomorphism called the *regular  $*$ -representation* of  $\mathcal{A}$ . The fact that  $L$  is linear and preserves matrix products is clear. Injectivity follows from the fact that  $L(A) = 0$  implies  $AA^* = 0$  and hence  $A = 0$ . Finally, the equations

$$L(A^*)_{st} = \langle A^* C_t, C_s \rangle = \langle C_t, AC_s \rangle = \overline{\langle AC_s, C_t \rangle} = \overline{L(A)_{ts}}$$

show that  $L(A^*) = L(A)^*$ . Because  $L$  is linear, it is determined by the images  $L(C_1), \dots, L(C_M)$ .

In many applications, for example in the case of a permutation action with the canonical basis  $C_1, \dots, C_M$  (Step 1 $\frac{1}{2}$  in the introduction), one only has an orthogonal basis which is not orthonormal. In that case, the map  $L$  is given by

$$L(C_r)_{st} = \frac{\langle C_r C_t, C_s \rangle}{\|C_s\| \|C_t\|} = \frac{\|C_s\|}{\|C_t\|} p_{rs}^t,$$

where the  $p_{rs}^t$  are the multiplication parameters defined by  $C_r C_s = \sum_t p_{rs}^t C_t$ . If we denote  $\phi(C_r) = L(C_r)$ , we obtain Theorem 9.2.

## 9.3 Invariant Positive Definite Functions on Compact Spaces

Until now we considered only finite dimensional invariant semidefinite programs and the question: How can symmetry coming from the action of a finite group be exploited to simplify the semidefinite program? In several situations, some given in Sect. 9.6, one wants to work with infinite dimensional invariant semidefinite programs. In these situations using the symmetry coming from the action of an infinite, continuous group is a must if one wants to do explicit computations. In this section we introduce the cone of continuous, positive definite functions on a compact set  $M$ , as a natural generalization of the cone of positive semidefinite matrices. When a compact group  $G$  acts continuously on  $M$ , we use the representation theory of  $G$  to describe the subcone of  $G$ -invariant positive definite functions on  $M$ . This is the

infinite dimensional analog of Step 2 (second version), given in the introduction, in the case of a permutation representation.

In general, this method evidences interesting links between the geometry of  $M$ , the representations of  $G$  and the theory of orthogonal polynomials. We will review the spaces  $M$ , finite and infinite, where this method has been completely worked out.

### 9.3.1 Positive Definite Functions

Let  $M$  be a compact space. We denote the space of continuous functions on  $M$  taking complex values by  $C(M)$ .

**Definition 9.4.** We say that  $F \in C(M^2)$  is *positive definite*,  $F \geq 0$ , if  $F(x, y) = \overline{F(y, x)}$  and, for all  $n$ , for all  $n$ -tuples  $(x_1, \dots, x_n) \in M^n$  and vectors  $(\alpha_1, \dots, \alpha_n) \in \mathbb{C}^n$ ,

$$\sum_{i,j=1}^n \alpha_i F(x_i, x_j) \overline{\alpha_j} \geq 0.$$

In other words, for all choices of finite subsets  $\{x_1, \dots, x_n\}$  of  $M$ , the matrix

$$(F(x_i, x_j))_{1 \leq i, j \leq n}$$

is Hermitian and positive semidefinite. The set of continuous positive definite functions on  $M$  is denoted  $C(M^2)_{\geq 0}$ .

In particular, if  $M$  is a finite set with the discrete topology, the elements of  $C(M^2)_{\geq 0}$  coincide with the Hermitian positive semidefinite matrices indexed by  $M$ . In general,  $C(M^2)_{\geq 0}$  is a closed convex cone in  $C(M^2)$ .

We now assume that a compact group  $G$  acts continuously on  $M$ . Here we mean that, in addition to the properties of a group action (see introduction), the map  $G \times M \rightarrow M$ , with  $(g, x) \mapsto gx$ , is continuous. We also assume that  $M$  is endowed with a *regular Borel measure*  $\mu$  which is  *$G$ -invariant*.

We recall here the basic notions on measures that will be needed. If  $M$  is a topological space, a *Borel measure* is a measure defined on the  $\sigma$ -algebra generated by the open sets of  $M$ . A *regular Borel measure* is one which behaves well with respect to this topology, we refer to Rudin [81] for the precise definition.

If  $M$  is a finite set, the topology on  $M$  is chosen to be the *discrete topology*, i.e. every subset of  $M$  is an open set, and the measure  $\mu$  on  $M$  is the *counting measure* defined by  $\mu(A) = |A|$ .

On a compact group  $G$ , there is a regular Borel measure  $\lambda$  which is *left and right invariant*, meaning that  $\lambda(gA) = \lambda(Ag) = \lambda(A)$  for all  $g \in G$  and  $A \subset G$  measurable, and such that the open sets have positive measure. This measure is unique up to a positive multiplicative constant and is called the *Haar measure on  $G$*  (see e.g. Conway [24, Theorem 11.4]). If  $G$  is finite, the counting measure on  $G$  provides a measure with these properties.

### 9.3.2 Unitary Representations of Compact Groups

The definitions and properties of representations given in the introduction extend naturally to compact groups. We only have to modify the definition of a finite dimensional representation by asking that the homomorphism  $\pi : G \rightarrow \mathrm{GL}(V)$  is continuous.

In the proofs, integrating with the *Haar measure*  $\lambda$  of  $G$  replaces averaging on finite groups. For example, in the introduction, we have seen that every representation  $V$  of a finite group  $G$  is unitary. The argument was, starting from an arbitrary inner product  $\langle u, v \rangle_0$  on  $V$ , to average it over all elements of  $G$ , in order to transform it into a  $G$ -invariant inner product. In the case of a more general compact group, we average over  $G$  using the Haar measure (normalized so that  $\lambda(G) = 1$ ), thus taking:

$$\langle u, v \rangle := \int_G \langle gu, gv \rangle d\lambda(g).$$

So Maschke's theorem holds: every finite dimensional  $G$ -space is the direct sum of irreducible  $G$ -subspaces. In contrast to the case of finite groups, the number of isomorphism classes of irreducible  $G$ -spaces is generally infinite. We fix a set  $\mathcal{R} = \{R_k : k \geq 0\}$  of representatives, assuming for simplicity that this set is countable (it is the case e.g. if  $G$  is a Lie group). Here  $R_0$  is the trivial representation, i.e.  $V = \mathbb{C}$  with  $gz = z$  for all  $g \in G, z \in \mathbb{C}$ .

However, infinite dimensional representations arise naturally in the study of infinite compact groups. We shall be primarily concerned with one of them, namely the representation of  $G$  defined by the space  $C(M)$  with the action of  $G$  defined by  $\pi(g)(f)(x) = f(g^{-1}x)$  (of course it has infinite dimension only if  $M$  itself is not a finite set). It is a unitary representation for the standard inner product on  $C(M)$ :

$$\langle f_1, f_2 \rangle = \frac{1}{\mu(M)} \int_M f_1(x) \overline{f_2(x)} d\mu(x).$$

Moreover  $C(M)$  is a dense subspace of the *Hilbert space*  $L^2(M)$  of measurable functions  $f$  on  $M$  such that  $|f|^2$  is  $\mu$ -integrable (see e.g. Conway [24, Chap. 1] or Rudin [81, Chap. 4]). It turns out that the theory of finite dimensional representations of compact groups extends nicely to the unitary representations on Hilbert spaces:

**Theorem 9.6.** *Every unitary representation of a compact group  $G$  on a Hilbert space is the direct sum (in the sense of Hilbert spaces) of finite dimensional irreducible  $G$ -subspaces.*

The above theorem is in fact a consequence of the celebrated *Theorem of Peter and Weyl* (see e.g. Bump [20]), which states that the matrix coefficients of finite dimensional irreducible representations of a compact group  $G$  span a subspace of  $C(G)$  which is dense for the topology of uniform convergence.

### 9.3.3 *G-Invariant Positive Definite Functions on M*

Now we come to our main concern, which is to describe the elements of  $C(M^2)_{\geq 0}^G$ , i.e. the  $G$ -invariant positive definite functions, in terms of the structure of the  $G$ -space  $M$ . In order to avoid technicalities with convergence issues, we shall systematically work in finite dimensional subspaces of  $C(M)$ . So let  $V$  be a  $G$ -subspace of  $C(M)$  of finite dimension. Let  $V^{(2)}$  denote the subspace of  $C(M^2)$  spanned by elements of the form  $f_1(x)\overline{f_2(y)}$ , where  $f_1$  and  $f_2$  belong to  $V$ .

We take the following notations for an irreducible decomposition of  $V$ :

$$V = \bigoplus_{k \in I_V} \bigoplus_{i=1}^{m_k} H_{k,i}, \quad (9.8)$$

where for  $1 \leq i \leq m_k$ , the subspaces  $H_{k,i}$  are pairwise orthogonal and  $G$ -isomorphic to  $R_k$ , and where  $I_V$  is the finite set of indices  $k \geq 0$  such that the multiplicities  $m_k$  are not equal to 0. For all  $k, i$ , we choose an orthonormal basis  $(e_{k,i,1}, \dots, e_{k,i,d_k})$  of  $H_{k,i}$ , where  $d_k = \dim(R_k)$ , such that the complex numbers  $\langle g e_{k,i,s}, e_{k,i,t} \rangle$  do not depend on  $i$  (such a basis exists precisely because the  $G$ -isomorphism class of  $H_{k,i}$  does not depend on  $i$ ). From this data, we introduce the  $m_k \times m_k$  matrices  $E_k(x, y)$  with coefficients  $E_{k,i,j}(x, y)$ :

$$E_{k,i,j}(x, y) = \sum_{s=1}^{d_k} e_{k,i,s}(x) \overline{e_{k,j,s}(y)}. \quad (9.9)$$

Then  $V^{(2)}$  obviously contains the elements  $e_{k,i,s}(x)\overline{e_{k,j,s}(y)}$ , which moreover form an orthonormal basis of this space.

**Theorem 9.7.** *Let  $F \in V^{(2)}$ . Then  $F$  is a  $G$ -invariant positive definite function if and only if there exist Hermitian positive semidefinite matrices  $F_k$  such that*

$$F(x, y) = \sum_{k \in I_V} \langle F_k, \overline{E_k(x, y)} \rangle. \quad (9.10)$$

Before we give a sketch of the proof, several comments are in order:

1. If  $M = [n]$ , then we recover the parametrization of  $G$ -invariant positive semidefinite matrices given in (9.7).
2. The main point of the above theorem is to replace the property of a continuous function being positive definite with the property that the  $m_k \times m_k$  matrices  $F_k$  are positive semidefinite. In the introduction, we have already seen in the finite case  $M = [n]$  that it allows to reduce the size of invariant semidefinite programs (following Step 2 (second version) of the introduction). In the general case, this theorem allows to replace a conic linear program involving the cone  $C(M^2)_{\geq 0}$  with standard semidefinite programs (see Sect. 9.6 for examples where  $M$  is the unit sphere of Euclidean space).

3. One might wonder to what extent the above matrices  $E_k(x, y)$  depend on the choices involved in their definition. Indeed, one can prove that a different choice of orthonormal basis in  $H_{k,i}$  does not affect  $E_k(x, y)$ . One could also start from another decomposition of the isotypic component  $\mathcal{M}I_k$  of  $V$ . The new decomposition does not need to be orthogonal; the matrix  $E_k(x, y)$  would then change to  $A E_k(x, y) A^*$  for some  $A \in \mathrm{GL}_{m_k}(\mathbb{C})$ . So, up to an action of  $\mathrm{GL}_{m_k}(\mathbb{C})$ , the matrix  $E_k(x, y)$  is canonically associated to  $\mathcal{M}I_k$ .
4. The statement implies that the matrices  $E_k(x, y)$  themselves are  $G$ -invariant. In other words, they can be expressed in terms of the orbits  $O(x, y)$  of the action of  $G$  on  $M^2$  in the form:

$$E_k(x, y) = Y_k(O(x, y)) \quad (9.11)$$

for some matrices  $Y_k$ . It is indeed the expression we are seeking for.

*Proof.*  $F(x, y)$  is a linear combination of the  $e_{k,i,s}(x)\overline{e_{l,j,t}(y)}$  since these elements form an orthonormal basis of  $V^{(2)}$ . In a first step, one shows that the  $G$ -invariance property  $F(gx, gy) = F(x, y)$  results in an expression for  $F$  of the form (9.10) for some matrices  $F_k = (f_{k,ij})$ ; the proof involves the *orthogonality relations* between matrix coefficients of irreducible representations of  $G$  (see e.g. Bump [20]). In a second step,  $F \geq 0$  is shown to be equivalent to  $F_k \geq 0$  for all  $k \in I_V$ . Indeed, for  $\alpha(x) = \sum_{i=1}^{m_k} \alpha_i e_{k,i,s}(x)$ , we have

$$\sum_{i,j=1}^{m_k} \alpha_i f_{k,ij} \overline{\alpha_j} = \frac{1}{\mu(M)^2} \int_{M^2} \alpha(x) F(x, y) \overline{\alpha(y)} d\mu(x, y) \geq 0.$$

□

*Remark 9.1.* 1. A straightforward generalization of the main structure theorem for matrix  $*$ -algebras to the  $*$ -algebra  $\mathrm{End}^G(V)$  shows that

$$\mathrm{End}^G(V) \simeq (V^{(2)})^G \simeq \bigoplus_{k \in I_V} \mathbb{C}^{m_k \times m_k}. \quad (9.12)$$

An isomorphism from  $(V^{(2)})^G$  to the direct sum of matrix algebras is constructed in the proof of Theorem 9.7, in an explicit way, from the decomposition (9.8). This isomorphism is given by the map sending  $(F_k)_{k \in I_V} \in \bigoplus_{k \in I_V} \mathbb{C}^{m_k \times m_k}$  to  $F \in (V^{(2)})^G$  defined by:

$$F(x, y) = \sum_{k \in I_V} \langle F_k, \overline{E_k(x, y)} \rangle$$

thus completing Step 2 (second version) of the introduction for compact groups.

2. Theorem 9.7 shows, jointly with Theorem 9.6, that any element of the cone  $C(M^2)_{\geq 0}^G$  is a sum (in the sense of  $L^2$  convergence) of the form (9.10) with possibly infinitely many terms. Moreover, under the additional assumption that  $G$  acts transitively on  $M$ , Bochner [16] proves that the convergence holds in the stronger sense of *uniform convergence*, i.e. for the supremum norm.

### 9.3.4 Examples: The Commutative Case

We mean here the case when  $\text{End}^G(C(M))$  is a commutative algebra. From Theorem 9.7, this condition is equivalent to the commutativity of  $\text{End}^G(V)$  for all finite dimensional  $G$ -subspace of  $C(M)$ . So, from the isomorphism (9.12), and since a matrix algebra  $\mathbb{C}^{m \times m}$  is commutative if and only if  $m = 0$  or  $m = 1$ , the commutative case corresponds to non vanishing multiplicities equal to 1 in the decomposition of  $C(M)$ . The  $G$ -space  $C(M)$  is said to be multiplicity-free. Then the matrices  $F_k$  in (9.10) have only one coefficient  $f_k$  and  $F \in C(M^2)_{\geq 0}^G$  is of the form

$$F(x, y) = \sum_{k \geq 0} f_k E_k(x, y), \quad f_k \geq 0. \quad (9.13)$$

We say that  $M$  is  *$G$ -symmetric*, if  $G$  acts transitively on  $M$  (i.e. there is only one orbit in  $M$ ; equivalently  $M$  is said to be a *homogeneous space*), and if moreover, for all  $(x, y) \in M^2$ , there exists  $g \in G$  such that  $gx = y$  and  $gy = x$ . In other words  $(y, x)$  belongs to the  $G$ -orbit  $O(x, y)$  of the pair  $(x, y)$ . This nonstandard terminology (sometimes it is also called a *generously transitive* group action) covers the case of *compact symmetric Riemannian manifolds* (see e.g. Berger [15]) as well as a large number of finite spaces. We provide many examples of such spaces below. The functions  $Y_k$  (9.11) are often referred to as the *spherical functions* or *zonal spherical functions* of the homogeneous space  $M$  (see e.g. Vilenkin and Klimyk [96]).

**Lemma 9.1.** *If  $M$  is  $G$ -symmetric then for all  $V \subset C(M)$ ,  $(V^{(2)})^G$  is commutative.*

*Proof.* Since  $F \in (V^{(2)})^G$  is  $G$ -invariant and  $M$  is  $G$ -symmetric,  $F(x, y) = F(y, x)$ . The multiplication on the algebra  $(V^{(2)})^G \simeq \text{End}^G(M)$  is the convolution  $F * F'$  of functions, given by:

$$(F * F')(x, y) = \int_M F(x, z) F'(z, y) d\mu(z).$$

A straightforward computation shows that  $F * F' = F' * F$ . □

An important subclass of symmetric spaces is provided by the *two-point homogeneous spaces*. These spaces are metric spaces, with distance  $d(x, y)$  invariant by  $G$ , and moreover satisfy that two pairs of elements  $(x, y)$  and  $(x', y')$  belong to the same  $G$ -orbit if and only if  $d(x, y) = d(x', y')$ . They are obviously  $G$ -symmetric since  $d(x, y) = d(y, x)$ . In other words, the distance function parametrizes the orbits of  $G$  acting on pairs, thus the expression (9.13) of positive definite functions specializes to

$$F(x, y) = \sum_{k \geq 0} f_k Y_k(d(x, y)), \quad f_k \geq 0, \quad (9.14)$$

for some functions  $Y_k$  in one variable. The functions  $Y_k$  are explicitly known for many spaces, and are usually given by a certain family of *orthogonal polynomials*

**Table 9.1** The real compact two-point homogeneous spaces, their groups and their spherical functions  $Y_k$

Space	Group	$\alpha$	$\beta$
$S^{n-1}$	$O_n(\mathbb{R})$	$(n-3)/2$	$(n-3)/2$
$P^{n-1}(\mathbb{R})$	$O_n(\mathbb{R})$	$(n-3)/2$	$-1/2$
$P^{n-1}(\mathbb{C})$	$U_n(\mathbb{C})$	$n-2$	$0$
$P^{n-1}(\mathbb{H})$	$U_n(\mathbb{H})$	$2n-3$	$1$
$P^2(\mathbb{O})$	$F_4(\mathbb{R})$	$7$	$3$

(see e.g. Szegö [89], Andrews et al. [1]). The orthogonality property arises because the associated irreducible subspaces are pairwise orthogonal.

The complete list of real, compact, connected two-point homogeneous spaces was established by Wang [97]: the unit sphere of Euclidean space  $S^{n-1} \subset \mathbb{R}^n$ , which is treated in Sect. 9.6, the projective spaces over the fields of real, complex and quaternion numbers (denoted  $P^{n-1}(K)$ , with respectively  $K = \mathbb{R}, \mathbb{C}, \mathbb{H}$ ), and the projective plane over the octonions  $P^2(\mathbb{O})$ , are the only spaces having these properties. In each of these cases the functions  $Y_k$  are given by *Jacobi polynomials* (see Table 9.1). These polynomials depend on two parameters  $(\alpha, \beta)$  and are orthogonal for the measure  $(1-t)^\alpha(1+t)^\beta dt$  on the interval  $[-1, 1]$ . We denote by  $P_k^{(\alpha, \beta)}(t)$  the Jacobi polynomial of degree  $k$ , normalized by the condition  $P_k^{(\alpha, \beta)}(1) = 1$ . When  $\alpha = \beta = \lambda - 1/2$ , these polynomials are equal (up to a nonnegative multiplicative factor) to the *Gegenbauer polynomials*  $C_k^\lambda(t)$ .

The finite two-point homogeneous spaces are not completely classified, but several of them are relevant spaces for coding theory. The most prominent one is the Hamming space  $\mathbf{q}^n$ , associated to the Krawtchouk polynomials, which is discussed in Sect. 9.4. It is the space of  $n$ -tuples, over a finite set  $\mathbf{q}$  of  $q$  elements. An element of the Hamming space  $\mathbf{q}^n$  is usually called a word, and  $\mathbf{q}$  is called the alphabet. The closely related *Johnson spaces* are the subsets of binary words of fixed length and weight. The Hamming and Johnson spaces are endowed with the *Hamming distance*, counting the number of coordinates where two words disagree. The Johnson spaces have  $q$ -analogues, the  *$q$ -Johnson spaces*, the subsets of linear subspaces of  $\mathbb{F}_q^n$  of fixed dimension. The distance between two subspaces of the same dimension is measured by the difference between this dimension and the dimension of their intersection. Other spaces are related to finite geometries and to spaces of matrices. In the later case, the distance between two matrices will be the rank of the difference of the matrices.

The spherical functions have been worked out, initially in the framework of association schemes with the work of Delsarte [28], and later in relation with harmonic analysis of finite classical groups with the work of Delsarte, Dunkl, Stanton and others. We refer to Conway and Sloane [25, Chap. 9] and to Stanton [86] for surveys on these topics. Table 9.2 displays the most important families of these spaces, together with their groups of isometries and the family of orthogonal polynomials to which their spherical functions belong; see Stanton [86] for additional information.

Symmetric spaces which are not two-point homogeneous give rise to functions  $Y_k$  depending on several variables. A typical example is provided by the real Grassmann spaces, the spaces of  $m$ -dimensional subspaces of  $\mathbb{R}^n$ ,  $m \leq n/2$ , under

**Table 9.2** Some finite two-point homogeneous spaces, their groups and their spherical functions  $Y_k$ 

Space	Group	Polynomial	Reference
Hamming space $\mathbf{q}^n$	$S_q \wr S_n$	Krawtchouk	[28]
Johnson space	$S_n$	Hahn	[28], [30]
$q$ -Johnson space	$\mathrm{Gl}_n(\mathbb{F}_q)$	$q$ -Hahn	[30]
<i>Maximal totally isotropic subspaces of dimension <math>k</math>, for a nonsingular bilinear form:</i>			
Symmetric	$\mathrm{SO}_{2k+1}(\mathbb{F}_q)$	$q$ -Krawtchouk	[86]
	$\mathrm{SO}_{2k}(\mathbb{F}_q)$	$q$ -Krawtchouk	[86]
	$\mathrm{SO}_{2k+2}^-(\mathbb{F}_q)$	$q$ -Krawtchouk	[86]
Symplectic	$\mathrm{Sp}_{2k}(\mathbb{F}_q)$	$q$ -Krawtchouk	[86]
Hermitian	$\mathrm{SU}_{2k}(\mathbb{F}_{q^2})$	$q$ -Krawtchouk	[86]
	$\mathrm{SU}_{2k+1}(\mathbb{F}_{q^2})$	$q$ -Krawtchouk	[86]
<i>Spaces of matrices:</i>			
$\mathbb{F}_q^{k \times n}$	$\mathbb{F}_q^{k \times n} \cdot (\mathrm{Gl}_k(\mathbb{F}_q) \times \mathrm{Gl}_n(\mathbb{F}_q))$	Affine $q$ -Krawtchouk	[31], [86]
Skew $_n(\mathbb{F}_q)$ skew-symmetric	Skew $_n(\mathbb{F}_q) \cdot \mathrm{Gl}_n(\mathbb{F}_q)$	Affine $q$ -Krawtchouk	[29], [86]
Herm $_n(\mathbb{F}_{q^2})$ Hermitian	Herm $_n(\mathbb{F}_{q^2}) \cdot \mathrm{Gl}_n(\mathbb{F}_{q^2})$	Affine $q$ -Krawtchouk	[86]

**Table 9.3** Some symmetric spaces, their groups and their spherical functions  $Y_k$ 

Space	Group	Polynomial	Reference
Nonbinary Johnson	$S_{q-1} \wr S_n$	Eberlein/Krawtchouk	[91]
Permutation group	$S_n^2$	Characters of $S_n$	[92]
Lee space	$D_q \wr S_n$	Multivariate Krawtchouk	[2]
Ordered Hamming space	$(\mathbb{F}_q^k \cdot B_k) \wr S_n$	Multivariate Krawtchouk	[69], [13]
Real Grassmann space	$O_n(\mathbb{R})$	Multivariate Jacobi	[48], [3]
Complex Grassmann space	$U_n(\mathbb{C})$	Multivariate Jacobi	[48], [79], [80]
Unitary group	$U_n(\mathbb{C})^2$	Schur	[26]

the action of the orthogonal group  $O(\mathbb{R}^n)$ . Then, the functions  $Y_k$  are multivariate Jacobi polynomials, with  $m$  variables representing the principal angles between two subspaces (see James and Constantine [48]). A similar situation occurs in coding theory when nonhomogeneous alphabets are considered, involving multivariate Krawtchouk polynomials. Table 9.3 shows some examples of these spaces with references to coding theory applications.

### 9.3.5 Examples: The Noncommutative Case

Only a few cases have been completely worked out. Among them, the Euclidean sphere  $S^{n-1}$  under the action of the subgroup of the orthogonal group fixing one point is treated in Bachoc and Vallentin [5]. We come back to this case in Sect. 9.6. These results have been extended to the case of fixing many points in Musin [73].

The case of the binary Hamming space under the action of the symmetric group is treated in the framework of group representations in Vallentin [94], shedding a new light on the results of Schrijver [84]. A very similar case is given by the set of linear subspaces of the finite vector space  $\mathbb{F}_q^n$ , treated as a  $q$ -analogue of the binary Hamming space in Bachoc and Vallentin [7].

## 9.4 Block Codes

In this section, we give an overview of symmetry reduction in semidefinite programming bounds for (error correcting) codes. We fix an alphabet  $\mathbf{q} = \{0, 1, \dots, q-1\}$  for some integer  $q \geq 2$ . The *Hamming space*  $\mathbf{q}^n$  is equipped with a metric  $d(\cdot, \cdot)$  called the *Hamming distance* which is given by

$$d(u, v) = |\{i : u_i \neq v_i\}| \text{ for all } u, v \in \mathbf{q}^n.$$

The isometry group of the Hamming space will be denoted  $\text{Aut}(q, n)$  and is a wreath product  $S_q \wr S_n$  of symmetric groups. That is, the isometries are obtained by taking a permutation of the  $n$  positions followed by independently permuting the symbols  $0, \dots, q-1$  at each of the  $n$  positions. The group acts transitively on  $\mathbf{q}^n$  and the orbits of pairs are given by the Hamming distance.

A subset  $C \subseteq \mathbf{q}^n$  is called a *code* of length  $n$ . For a (nonempty) code  $C$ ,  $\min\{d(u, v) : u, v \in C \text{ distinct}\}$  is the *minimum distance* of  $C$ . An important quantity in coding theory is the maximum size  $A_q(n, d)$  of a code of length  $n$  and minimum distance at least  $d$ :

$$A_q(n, d) = \max\{|C| : C \subseteq \mathbf{q}^n \text{ has minimum distance at least } d\}.$$

These codes are often called *block codes* since they can be used to encode messages into fixed-length blocks of words from a code  $C$ . The most studied case is that of *binary codes*, that is  $q = 2$  and  $\mathbf{q} = \{0, 1\}$ . In this case  $\mathbf{q}$  is suppressed from the notation. An excellent reference work on coding theory is MacWilliams and Sloane [68].

Lower bounds for  $A_q(n, d)$  are mostly obtained using explicit construction of codes. Our focus here is on upper bounds obtained using semidefinite programming.

### 9.4.1 Lovász' $\vartheta$ and Delsarte's Linear Programming Bound

Let  $\Gamma_q(n, d)$  be the graph on vertex set  $\mathbf{q}^n$ , connecting two words  $u, v \in \mathbf{q}^n$  if  $d(u, v) < d$ . Then the codes of minimum distance at most  $d$  correspond exactly to

the independent sets in  $\Gamma_q(n, d)$  and hence

$$A_q(n, d) = \alpha(\Gamma_q(n, d)),$$

where  $\alpha$  denotes the independence number of a graph. The graph parameter  $\vartheta'$ , defined in Schrijver [84], is a slight modification of the Lovász theta number discussed in more detail in Sect. 9.6. It can be defined by the following semidefinite program

$$\vartheta'(\Gamma) = \max\{\langle X, J \rangle : \langle X, I \rangle = 1, X_{uv} = 0 \text{ if } uv \in E(\Gamma), X \geq 0, X \geq 0\},$$

and it is a by-now classical lemma of Lovász [67] that for any graph  $\Gamma$  the inequality  $\alpha(\Gamma) \leq \vartheta'(\Gamma) \leq \vartheta(\Gamma)$  holds.

Hence  $\vartheta'(\Gamma_q(n, d))$  is an (exponential size) semidefinite programming upper bound for  $A_q(n, d)$ . Using symmetry reduction, this program can be solved efficiently. For this, observe that the SDP is invariant under the action of  $\text{Aut}(q, n)$ . Hence we may restrict  $X$  to belong to the matrix  $*$ -algebra  $\mathcal{A}$  of invariant matrices (the Bose–Mesner algebra of the Hamming scheme). The zero-one matrices  $A_0, \dots, A_n$  corresponding to the orbits of pairs:

$$(A_i)_{u,v} = \begin{cases} 1 & \text{if } d(u, v) = i, \\ 0 & \text{otherwise,} \end{cases}$$

form the canonical basis of  $\mathcal{A}$ . Writing  $X = \frac{1}{q^n}(x_0 A_0 + \dots + x_n A_n)$ , we obtain an SDP in  $n+1$  variables:

$$\begin{aligned} \vartheta'(\Gamma_q(n, d)) = \max \Bigg\{ & \sum_{i=0}^n x_i \binom{n}{i} : x_0 = 1, x_1 = \dots = x_{d-1} = 0, \\ & x_d, \dots, x_n \geq 0, \sum_{i=0}^n x_i A_i \succeq 0 \Bigg\}. \end{aligned}$$

The second step is to block diagonalize the algebra  $\mathcal{A}$ . In this case  $\mathcal{A}$  is commutative, which means that the  $A_i$  can be simultaneously diagonalized, reducing the positive semidefinite constraint to the linear constraints

$$\sum_{i=0}^n x_i P_i(j) \geq 0 \text{ for } j = 0, \dots, n,$$

in the eigenvalues  $P_i(j)$  of the matrices  $A_i$ . The  $P_i$  are the *Krawtchouk polynomials* and are given by

$$P_i(x) = \sum_{k=0}^i (-1)^k \binom{x}{k} \binom{n-x}{i-k} (q-1)^{i-k}.$$

Thus the semidefinite program is reduced to a linear program, which is precisely the *Delsarte bound* [28]. This relation between  $\vartheta'$  and the Delsarte bound was recognized independently in McEliece et al. [70] and Schrijver [83]. In the more general setting of (commutative) association schemes, Delsarte's linear programming bound is obtained from a semidefinite program by symmetry reduction and diagonalizing the Bose–Mesner algebra.

#### 9.4.2 Stronger Bounds Through Triples

Delsarte's bound is based on the distance distribution of a code, that is, on the number of code word-pairs for each orbit of pairs. Using orbits of triples, the bound can be tightened as was shown in Schrijver [84]. We describe this method for the binary case ( $q = 2$ ).

Denote by  $\text{Stab}(0, \text{Aut}(2, n)) \subseteq \text{Aut}(2, n)$  the stabilizer of 0. That is,  $\text{Stab}(0, \text{Aut}(2, n))$  consists of just the permutations of the  $n$  positions.

For any  $u, v, w \in \{0, 1\}^n$ , denote by  $O(u, v, w)$  the orbit of the triple  $(u, v, w)$  under the action of  $\text{Aut}(2, n)$ . Observe that since  $\text{Aut}(2, n)$  acts transitively on  $\{0, 1\}^n$ , the orbits of triples are in bijection with the orbits of pairs under  $\text{Stab}(0, \text{Aut}(2, n))$ , since we may assume that  $u$  is mapped to 0. There are  $\binom{n+3}{3}$  such orbits, indexed by integers  $0 \leq t \leq i, j \leq n$ . Indeed, the orbit of  $(v, w)$  under  $\text{Stab}(0, \text{Aut}(2, n))$  is determined by the sizes  $i, j$  and  $t$  of respectively the supports of  $v$  and  $w$  and their intersection.

Let  $C \subseteq \{0, 1\}^n$  be a code. We denote by  $\lambda_{i,j}^t$  the number of triples  $(u, v, w) \in C^3$  in the orbit indexed by  $i, j$  and  $t$ . This is the analogue for triples of the distance distribution.

Denote by  $M_C$  the zero-one  $\{0, 1\}^n \times \{0, 1\}^n$  matrix defined by  $(M_C)_{u,v} = 1$  if and only if  $u, v \in C$ . So  $M_C$  is a rank 1 positive semidefinite matrix. Now consider the following two matrices

$$M' = \sum_{\sigma \in \text{Aut}(2, n), 0 \in \sigma C} M_{\sigma C}, \quad M'' = \sum_{\sigma \in \text{Aut}(2, n), 0 \notin \sigma C} M_{\sigma C}.$$

Clearly,  $M'$  and  $M''$  are nonnegative and positive semidefinite. Furthermore, by construction  $M'$  and  $M''$  are invariant under the action of the stabilizer  $\text{Stab}(0, \text{Aut}(2, n))$ . Equivalently, any entry  $M'_{uw}$  (or  $M''_{uw}$ ) only depends on the orbit of  $(0, u, v)$  under the action of  $\text{Aut}(2, n)$ . In fact,  $M'_{uw}$  equals (up to a factor depending on the orbit) the number of triples  $(c, c', c'') \in C^3$  in that orbit.

The matrix  $*$ -algebra  $\mathcal{A}_n$  of complex  $\{0, 1\}^n \times \{0, 1\}^n$  matrices invariant under  $\text{Stab}(0, \text{Aut}(2, n))$  has a basis of zero-one matrices corresponding to the orbits of pairs under the action of  $\text{Stab}(0, \text{Aut}(2, n))$ :  $\{A_{i,j}^t : 0 \leq t \leq i, j \leq n\}$ . This algebra is called the *Terwilliger algebra* of the (binary) Hamming scheme. The facts about  $M'$  and  $M''$  above lead to a semidefinite programming bound for  $A(n, d)$ , in terms of variables  $x_{i,j}^t$ . Nonnegativity of  $M', M''$  translates into nonnegativity of the  $x_{i,j}^t$ . Excluding positive distances smaller than  $d$  translates into setting  $x_{i,j}^t = 0$  for orbits

containing two words at distance 1 through  $d - 1$ . Semidefiniteness of  $M'$  and  $M''$  translate into

$$\sum_{i,j,t} x_{i,j}^t A_{i,j}^t \geq 0, \quad \sum_{i,j,t} (x_{i+j-t,0}^0 - x_{i,j}^t) A_{i,j}^t \geq 0.$$

There are some more constraints. In particular, for  $u, v, w \in \{0, 1\}^n$  the variables of the orbits of  $(u, v, w)$  and  $(w, u, v)$  (or any other reordering) must be equal. This further reduces the number of variables. For more details, see Schrijver [84].

Having reduced the SDP-variables by using the problem symmetry, the second step is to replace the positive semidefinite constraints by equivalent conditions on smaller matrices using a block diagonalization of  $\mathcal{A}_n$ . The block diagonalization of  $\mathcal{A}_n$  is given explicitly in Schrijver [84] (see also Vallentin [94], Dunkl [35]). The number of blocks equals  $1 + \lfloor \frac{n}{2} \rfloor$ , with sizes  $n + 1 - 2k$  for  $k = 0, \dots, \lfloor \frac{n}{2} \rfloor$ . Observe that the sum of the squares of these numbers indeed equals  $\binom{n+3}{3}$ , the dimension of  $\mathcal{A}_n$ .

For the non-binary case, an SDP upper bound for  $A_q(n, d)$  can be defined similarly. The main differences are, that the orbits or triples are now indexed by four parameters  $i, j, p, t$  satisfying  $0 \leq p \leq t \leq i, j$ ,  $i + j - t \leq n$ . Hence the corresponding invariant algebra is a matrix  $*$ -algebra of dimension  $\binom{n+4}{4}$ . In that case the block diagonalization is a bit more involved having blocks indexed by the integers  $a, k$  with  $0 \leq a \leq k \leq n + a - k$ , and size  $n + a + 1 - 2k$ . See Gijswijt et al. [41] for details.

### 9.4.3 Hierarchies and $k$ -Tuples

#### 9.4.3.1 Moment Matrices

Let  $V$  be a finite set and denote by  $\mathcal{P}(V)$  its power set. For any  $y : \mathcal{P}(V) \rightarrow \mathbb{C}$ , denote by  $M(y)$  the  $\mathcal{P}(V) \times \mathcal{P}(V)$  matrix given by

$$[M(y)]_{S,T} = y_{S \cup T}.$$

Such a matrix is called a *combinatorial moment matrix* (see Laurent [64]).

Let  $C \subseteq V$  and denote by  $x = \chi^C \in \{0, 1\}^V$  the zero-one incidence vector of  $C$ . The vector  $x$  can be “lifted” to  $y \in \{0, 1\}^{\mathcal{P}(V)}$  by setting  $y_S = \prod_{v \in S} x_v$  for all  $S \in \mathcal{P}(V)$ . Note that  $y_\emptyset = 1$ . Then the moment matrix  $M(y)$  is positive semidefinite since  $M(y) = yy^\top$ . This observation has an important converse (see Laurent [64]).

**Theorem 9.8.** *Let  $M(y)$  be a moment matrix with  $y_\emptyset = 1$  and  $M(y) \succeq 0$ . Then  $M(y)$  is a convex combination of moment matrices  $M(y_1), \dots, M(y_k)$ , where  $y_i$  is obtained from lifting a vector in  $\{0, 1\}^V$ .*

Let  $\Gamma = (V, E)$  be a graph on  $V$ . Theorem 9.8 shows that the independence number is given by

$$\alpha(\Gamma) = \max \left\{ \sum_{v \in V} y_{\{v\}} : y_\emptyset = 1, y_S = 0 \text{ if } S \text{ contains an edge,} \right. \\ \left. M(y) \geq 0 \right\}. \quad (9.15)$$

By replacing  $M(y)$  by suitable principal submatrices, this (exponential size) semidefinite program can be relaxed to obtain more tractable upper bounds on  $\alpha(\Gamma)$ . For example, restricting rows and columns to sets of size at most 1 (and restricting  $y$  to sets of size  $\leq 2$ ), we obtain the Lovász  $\vartheta$  number (adding nonnegativity of  $y_S$  for  $S$  of size 2 gives  $\vartheta'$ ).

To describe a large class of useful submatrices, fix a nonnegative integer  $s$  and a set  $T \subseteq V$ . Define  $M_{s,T} = M_{s,T}(y)$  to be the matrix with rows and columns indexed by sets  $S$  of size at most  $s$ , defined by

$$[M_{s,T}]_{S,S'} = y_{S \cup S' \cup T}. \quad (9.16)$$

So  $M_{s,T}$  is a principal submatrix of  $M(y)$ , except that possibly some rows and columns are duplicated. For fixed  $k$ , we can restrict  $y$  to subsets of size at most  $k$  and replace  $M(y) \geq 0$  in (9.15) by conditions

$$M_{s,T} \geq 0 \quad \text{for all } T \text{ of size at most } t,$$

where  $2s + t \leq k$ . This yields upper bounds on  $\alpha(\Gamma)$  defined in terms of subsets of  $V$  of size at most  $k$ . The constraints (9.16) can be strengthened further to

$$\tilde{M}_{s,T} = (M_{s,T' \cup T''})_{T', T'' \subseteq T} \geq 0 \quad \text{for all } T \text{ of size } t. \quad (9.17)$$

Here  $\tilde{M}_{s,T}$  is a  $2^{|T|} \times 2^{|T|}$  matrix of smaller matrices, which can be seen as a submatrix of  $M(y)$  (again up to duplication of rows and columns). By the moment-structure of  $\tilde{M}_{s,T}$ , condition (9.17) is equivalent to

$$\sum_{R' \subseteq R} (-1)^{|R \setminus R'|} M_{s,R'} \geq 0 \quad \text{for all } R \text{ of size at most } t.$$

Using SDP constraints of the form (9.16) and (9.17) with  $2s + t \leq k$ , we can obtain the (modified) theta number ( $k = 2$ ), Schrijver's bound using triples ( $k = 3$ ), or rather a strengthened version from Laurent [63], and the bound from Gijswijt et al. [40] ( $k = 4$ ).

Several hierarchies of upper bounds have been proposed. The hierarchy introduced by Lasserre [62] (also see Laurent [63]) is obtained using (9.16) by fixing

$t = 0$  and letting  $s = 1, 2, \dots$  run over the positive integers. The hierarchy described in Gijswijt, Mittelmann, Schrijver [40] is obtained using (9.16) by letting  $k = 1, 2, \dots$  run over the positive integers and restricting  $s$  and  $t$  by  $2s + t \leq k$ . Finally, the hierarchy defined in Gvozdenović et al. [46] fixes  $s = 1$  and considers the constraints (9.17), where  $t = 0, 1, \dots$  runs over the nonnegative integers. There, also computational results are obtained for Paley graphs in the case  $t = 1, 2$ .

A slight variation is obtained by restricting  $y$  to sets of at least one element, deleting the row and column corresponding to the empty set from (submatrices of)  $M(y)$ . The normalization  $y_{\emptyset} = 1$  is then replaced by  $\sum_{v \in V} y_{\{v\}} = 1$  and the objective is replaced by  $\sum_{u, v \in V} y_{\{u, v\}}$ . Since  $M_{1, \emptyset} \geq 0$  implies that  $(\sum_{v \in V} y_{\{v\}})^2 \leq y_{\emptyset} \sum_{u, v \in V} y_{\{u, v\}}$ , this gives a relaxation. This variation was used in Schrijver [84] and Gijswijt et al. [41].

#### 9.4.3.2 Symmetry Reduction

Application to the graph  $\Gamma = \Gamma_q(n, d)$  with vertex set  $V = \mathbf{q}^n$ , gives semidefinite programming bounds for  $A_q(n, d)$ . A priori, this gives an SDP that has exponential size in  $n$ . However, using the symmetries of the Hamming space, it can be reduced to polynomial size as follows.

The action of  $\text{Aut}(q, n)$  on the Hamming space  $V$ , extends to an action on  $\mathcal{P}(V)$  and hence on vectors and matrices indexed by  $\mathcal{P}(V)$ . Since the semidefinite program in (9.15) is invariant under  $\text{Aut}(q, n)$ , we may restrict  $y$  (and hence  $M(y)$ ) to be invariant under  $\text{Aut}(q, n)$ , without changing the optimum value. The same holds for any of the bounds obtained using principal submatrices. Thus  $M(y)_{S, T}$  only depends on the orbit of  $S \cup T$ . In particular, for any fixed integer  $k$ , the matrices  $M_{s, T}$  with  $2s + |T| \leq k$  are described using variables corresponding to the orbits of sets of size at most  $k$ . Two matrices  $M_{s, T}$  and  $M_{s, T'}$  are equal (up to permutation of rows and columns) when  $T$  and  $T'$  are in the same orbit under  $\text{Aut}(q, n)$ . Hence, the number of variables is bounded by the number of orbits of sets of size at most  $k$  ( $O(n^{2^{k-1}-1})$  in the binary case), and for each such orbit there are at most a constant number of distinct constraints of the form (9.16, 9.17). This concludes the first step in the symmetry reduction.

The matrices  $M_{s, T}$  are invariant under the subgroup of  $\text{Stab}(T, \text{Aut}(q, n))$  that fixes each of the elements of  $T$ . The dimension of the matrix  $*$ -algebra of invariant matrices is polynomial in  $n$  and hence the size of the matrices can be reduced to polynomial size using the regular  $*$ -representation.

By introducing more duplicate rows and columns and disregarding the row and column indexed by the empty set, we may index the matrices  $M_{s, T}$  by *ordered*  $s$ -tuples and view  $M_{s, T}$  as an  $(\mathbf{q}^n)^s \times (\mathbf{q}^n)^s$  matrix. Let  $\mathcal{A}_{s, T}$  be the matrix  $*$ -algebra of  $(\mathbf{q}^n)^s \times (\mathbf{q}^n)^s$  matrices invariant under the subgroup of  $\text{Stab}(T, \text{Aut}(q, n))$ . This matrix  $*$ -algebra can be block diagonalized using techniques from Gijswijt [39].

#### 9.4.4 Generalizations

The semidefinite programs discussed in the previous sections, are based on the distribution of pairs, triples and  $k$ -tuples in a code. For each tuple (up to isometry) there is a variable that reflects the number of occurrences of that tuple in the code. Exclusion of pairs at distance  $1, \dots, d-1$ , is modeled by setting variables for violating tuples to zero.

Bounds for other types of codes in the Hamming space can be obtained by setting the variables for excluded tuples to zero. This does not affect the underlying algebra of invariant matrices or the symmetry reduction. For triples in the binary Hamming space, this method was applied to orthogonality graphs in de Klerk and Pasechnik [57] and to pseudo-distances in Bachoc and Zémor [9]. Lower bounds on (nonbinary) covering codes have been obtained in Gijswijt [38] by introducing additional linear constraints.

Bounds on constant weight binary codes were given in Schrijver [84].

In the nonbinary Hamming space, the symbols  $0, \dots, q-1$  are all interchangeable. In the case of Lee-codes, the dihedral group  $D_q$  acts on the alphabet, which leads to a smaller isometry group  $D_q \wr S_n$ . The Bose–Mesner algebra of the Lee-scheme is still commutative. The corresponding linear programming bound was studied in Astola [2]. To the best knowledge of the author, stronger bounds using triples have not been studied in this case.

## 9.5 Crossing Numbers

We describe an application of the regular  $*$ -representation to obtain a lower bound on the crossing number of complete bipartite graphs. This was described in de Klerk et al. [58], and extends a method of de Klerk et al. [54].

The *crossing number*  $\text{cr}(\Gamma)$  of a graph  $\Gamma$  is the minimum number of intersections of edges when  $\Gamma$  is drawn in the plane such that all vertices are distinct. The *complete bipartite graph*  $K_{m,n}$  is the graph with vertices  $1, \dots, m, u_1, \dots, u_n$  and edges all pairs  $iu_j$  for  $i \in [m]$  and  $j \in [n]$ . (This notation will be convenient for our purposes.)

This relates to the problem raised by the paper of Zarankiewicz [99], asking if

$$\text{cr}(K_{m,n}) \stackrel{?}{=} Z(m, n) = \lfloor \frac{1}{4}(m-1)^2 \rfloor \lfloor \frac{1}{4}(n-1)^2 \rfloor. \quad (9.18)$$

In fact, Zarankiewicz claimed to have a proof, which however was shown to be incorrect. In (9.18),  $\leq$  follows from a direct construction. Equality was proved by Kleitman [52] if  $\min\{m, n\} \leq 6$  and by Woodall [98] if  $m \in \{7, 8\}$  and  $n \in \{7, 8, 9, 10\}$ .

Consider any  $m, n$ . Let  $Z_m$  be the set of cyclic permutations of  $[m]$  (that is, the permutations with precisely one orbit). For any drawing of  $K_{m,n}$  in the plane and for any  $u_i$ , let  $\gamma(u_i)$  be the cyclic permutation  $(1, i_2, \dots, i_m)$  such that the edges leaving  $u_i$  in clockwise order, go to  $1, i_2, \dots, i_m$  respectively.

For  $\sigma, \tau \in Z_m$ , let  $C_{\sigma, \tau}$  be equal to the minimum number of crossings when we draw  $K_{m,2}$  in the plane such that  $\gamma(u_1) = \sigma$  and  $\gamma(u_2) = \tau$ . This gives a matrix  $C = (C_{\sigma, \tau})$  in  $\mathbb{R}^{Z_m \times Z_m}$ . Then the number  $\alpha_m$  is defined by:

$$\alpha_m = \min\{\langle X, C \rangle : X \in \mathbb{R}_+^{Z_m \times Z_m}, X \geq 0, \langle X, J \rangle = 1\}, \quad (9.19)$$

where  $J$  is the all-one matrix in  $\mathbb{R}^{Z_m \times Z_m}$ .

Then  $\alpha_m$  gives a lower bound on  $\text{cr}(K_{m,n})$ , as was shown by de Klerk, et. al. [54].

**Theorem 9.9.**  $\text{cr}(K_{m,n}) \geq \frac{1}{2}n^2\alpha_m - \frac{1}{2}n\lfloor\frac{1}{4}(m-1)^2\rfloor$  for all  $m, n$ .

*Proof.* Consider a drawing of  $K_{m,n}$  in the plane with  $\text{cr}(K_{m,n})$  crossings. For each cyclic permutation  $\sigma$ , let  $d_\sigma$  be the number of vertices  $u_i$  with  $\gamma(u_i) = \sigma$ . Consider  $d$  as column vector in  $\mathbb{R}^{Z_m}$ , and define the matrix  $X$  in  $\mathbb{R}^{Z_m \times Z_m}$  by

$$X = n^{-2}dd^\top.$$

Then  $X$  satisfies the constraints in (9.19), hence  $\alpha_m \leq \langle X, C \rangle$ . For  $i, j = 1, \dots, n$ , let  $\beta_{i,j}$  denote the number of crossings of the edges leaving  $u_i$  with the edges leaving  $u_j$ . So if  $i \neq j$ , then  $\beta_{i,j} \geq C_{\gamma(u_i), \gamma(u_j)}$ . Hence

$$\begin{aligned} n^2\langle X, C \rangle &= \langle dd^\top C \rangle = d^\top Cd = \sum_{i,j=1}^n C_{\gamma(u_i), \gamma(u_j)} \\ &\leq \sum_{\substack{i,j=1 \\ i \neq j}}^n \beta_{i,j} + \sum_{i=1}^n C_{\gamma(u_i), \gamma(u_i)} = 2\text{cr}(K_{m,n}) + n\lfloor\frac{1}{4}(m-1)^2\rfloor. \end{aligned}$$

Therefore,

$$\text{cr}(K_{m,n}) \geq \frac{1}{2}n^2\langle X, C \rangle - \frac{1}{2}n\lfloor\frac{1}{4}(m-1)^2\rfloor \geq \frac{1}{2}\alpha_m n^2 - \frac{1}{2}n\lfloor\frac{1}{4}(m-1)^2\rfloor. \quad \square$$

The semidefinite programming problem (9.19) defining  $\alpha_m$  has order  $(m-1)!$ . Using the regular  $*$ -representation, we can reduce the size. Fix  $m \in \mathbb{N}$ . Let  $G = S_m \times \{-1, +1\}$ , and define  $h : G \rightarrow S_{Z_m}$  by

$$h_{\pi,i}(\sigma) = \pi\sigma^i\pi^{-1}$$

for  $\pi \in S_m$ ,  $i \in \{-1, +1\}$ ,  $\sigma \in Z_m$ . So  $G$  acts on  $Z_m$ . It is easy to see that  $C$  and (trivially)  $J$  are  $G$ -invariant.

Using the regular  $*$ -representation,  $\alpha_m$  up to  $m = 9$  was computed. For  $m = 8$ ,  $\alpha_8 = 5.8599856444\dots$ , implying

$$\text{cr}(K_{8,n}) \geq 2.9299n^2 - 6n.$$

This implies for each fixed  $m \geq 8$ , with an averaging argument over all subgraphs  $K_{8,n}$ :

$$\lim_{n \rightarrow \infty} \frac{\text{cr}(K_{m,n})}{Z(m,n)} \geq 0.8371 \frac{m}{m-1}.$$

Moreover,  $\alpha_9 = 7.7352126\dots$ , implying

$$\text{cr}(K_{9,n}) \geq 3.8676063n^2 - 8n,$$

and for each fixed  $m \geq 9$ :

$$\lim_{n \rightarrow \infty} \frac{\text{cr}(K_{m,n})}{Z(m,n)} \geq 0.8594 \frac{m}{m-1}.$$

Thus we have an asymptotic approximation to Zarankiewicz's problem.

The orbits of the action of  $G$  onto  $Z_m \times Z_m$  can be identified as follows. Each orbit contains an element  $(\sigma_0, \tau)$  with  $\sigma_0 = (1, \dots, m)$ . So there are at most  $(m-1)!$  orbits. Next,  $(\sigma_0, \tau)$  and  $(\sigma_0, \tau')$  belong to the same orbit if and only if  $\tau' = \tau^g$  for some  $g \in G$  that fixes  $\sigma_0$ . (There are only few of such  $g$ .) In this way, the orbits can be identified by computer, for  $m$  not too large. The corresponding values  $C_{\sigma,\tau}$  for each orbit  $[\sigma, \tau]$ , and the multiplication parameters, also can be found using elementary combinatorial algorithms. The computer does this all within a few minutes, for  $m \leq 9$ . But the resulting semidefinite programming problem took, in 2006, seven days. It was the largest SDP problem solved by then.

## 9.6 Spherical Codes

Finding upper bounds for codes on the sphere is our next application. Here one deals with infinite-dimensional semidefinite programs which are invariant under the orthogonal group which is continuous and compact. So we are in the situation for which the techniques of Sect. 9.3 work.

We start with some definitions: The unit sphere  $S^{n-1}$  of the Euclidean space  $\mathbb{R}^n$  equipped with its standard inner product  $x \cdot y = \sum_{i=1}^n x_i y_i$ , is defined as usual by:

$$S^{n-1} = \{(x_1, \dots, x_n) \in \mathbb{R}^n : x \cdot x = 1\}.$$

It is a compact space, on which the orthogonal group  $O_n(\mathbb{R})$  acts homogeneously. The stabilizer  $\text{Stab}(x_0, O_n(\mathbb{R}))$  of one point  $x_0 \in S^{n-1}$  can be identified with the orthogonal group of the orthogonal complement  $(\mathbb{R}x_0)^\perp$  of the line  $\mathbb{R}x_0$  thus with  $O_{n-1}(\mathbb{R})$ , leading to an identification between the sphere and the quotient space  $O_n(\mathbb{R})/O_{n-1}(\mathbb{R})$ . The unit sphere is endowed with the standard  $O_n(\mathbb{R})$ -invariant Lebesgue measure  $\mu$  with the normalization  $\mu(S^{n-1}) = 1$ . The *angular distance*  $d_\theta(x, y)$  of  $(x, y) \in (S^{n-1})^2$  is defined by

$$d_\theta(x, y) = \arccos(x \cdot y)$$

and is  $O_n(\mathbb{R})$ -invariant. Moreover, the metric space  $(S^{n-1}, d_\theta)$  is *two-point homogeneous* (see Sect. 9.3.4) under  $O_n(\mathbb{R})$ .

The *minimal angular distance*  $d_\theta(C)$  of a subset  $C \subset S^{n-1}$  is by definition the minimal angular distance of pairs of distinct elements of  $C$ . It is a classical problem to determine the maximal size of  $C$ , subject to the condition that  $d_\theta(C)$  is greater or equal to some given minimal value  $\theta_{\min}$ . This problem is the fundamental question of the theory of error correcting codes (see e.g. Conway and Sloane [25], Ericson and Zinoviev [36]). In this context, the subsets  $C \subset S^{n-1}$  are referred to as *spherical codes*. In geometry, the case  $\theta_{\min} = \pi/3$  corresponds to the famous *kissing number problem* which asks for the maximal number of spheres that can simultaneously touch a central sphere without overlapping, all spheres having the same radius (see e.g. Conway and Sloane [25]).

We introduce notations in a slightly more general framework. We say that  $C \subset S^{n-1}$  avoids  $\mathcal{Q} \subset (S^{n-1})^2$  if, for all  $(x, y) \in C^2$ ,  $(x, y) \notin \mathcal{Q}$ . We define, for a measure  $\lambda$ ,

$$A(S^{n-1}, \mathcal{Q}, \lambda) = \sup \{ \lambda(C) : C \subset S^{n-1} \text{ measurable, } C \text{ avoids } \mathcal{Q} \}.$$

We have in mind the following situations of interest:

1.  $\mathcal{Q} = \{(x, y) : d_\theta(x, y) \in ]0, \theta_{\min}[ \}$  and  $\lambda$  is the counting measure denoted  $\mu_c$ . Then, the  $\mathcal{Q}$ -avoiding sets are exactly the spherical codes  $C$  such that  $d_\theta(C) \geq \theta_{\min}$ .
2.  $\mathcal{Q} = \{(x, y) : d_\theta(x, y) = \theta \}$  for some value  $\theta \neq 0$ , and  $\lambda = \mu$ . Here we consider subsets avoiding only one value of distance, so these subsets can be infinite. This case has interesting connections with the famous problem of finding the *chromatic number of Euclidean space* (see Soifer [85], Bachoc et al. [10], Oliveira and Vallentin [77], Oliveira [76])
3.  $\mathcal{Q} = \{(x, y) : d_\theta(x, y) \in ]0, \theta_{\min}[ \text{ or } (x, y) \notin \text{Cap}(e, \phi)^2 \}$ , and  $\lambda = \mu_c$ . Here  $\text{Cap}(e, \phi)$  denotes the *spherical cap* with center  $e$  and radius  $\phi$ :

$$\text{Cap}(e, \phi) = \{x \in S^{n-1} : d_\theta(e, x) \leq \phi\}$$

and we are dealing with subsets of a spherical cap with given minimal angular distance.

The computation of  $A(S^{n-1}, \mathcal{Q}, \lambda)$  is a difficult and unsolved problem in general, so one aims at finding lower and upper bounds for this number. To that end, for finding upper bounds, we borrow ideas from combinatorial optimization. Indeed, if one thinks of the pair  $(S^{n-1}, \mathcal{Q})$  being a graph with vertex set  $S^{n-1}$  and edge set  $\mathcal{Q}$ , then a set  $C$  avoiding  $\mathcal{Q}$  corresponds to an *independent set*, and  $A(S^{n-1}, \mathcal{Q}, \lambda)$  to the *independence number* of this graph. In general, for a graph  $\Gamma$  with vertex set  $V$  and edge set  $E$ , an *independent set* is a subset  $S$  of  $V$  such that no pairs of elements of  $S$  are connected by an edge, and the *independence number*  $\alpha(\Gamma)$  of  $\Gamma$  is the maximal number of elements of an independent set. The *Lovász theta number*  $\vartheta(\Gamma)$  was introduced by Lovász [67]. It gives an upper bound of the independence number  $\alpha(\Gamma)$  of a graph  $\Gamma$ , and it is the optimal solution of a semidefinite program. We review Lovász theta number in Sect. 9.6.1. Then, in Sect. 9.6.2, we introduce generalizations of this notion, in the form of conic linear

programs, that provide upper bounds for  $A(S^{n-1}, \mathcal{Q}, \lambda)$ . Using symmetry reduction, in the cases (i), (ii), (iii) above, it is possible to approximate these conic linear programs with semidefinite programs, that can be practically computed when the dimension  $n$  is not too large. This step is explained in Sect. 9.6.5. It involves the description of the cones of  $G$ -invariant positive definite functions, for the groups  $G = O_n(\mathbb{R})$  and  $G = \text{Stab}(x_0, O_n(\mathbb{R}))$ . We provide this description, following the lines of Sect. 9.3, in Sects. 9.6.3 and 9.6.4. Finally, in Sect. 9.6.6 we indicate further applications.

### 9.6.1 Lovász $\vartheta$

This number can be defined in many equivalent ways (see Lovász [67], Knuth [61]). We present here the most appropriate one in view of our purpose, which is the generalization to  $S^{n-1}$ .

**Definition 9.5.** The *theta number* of the graph  $\Gamma = (V, E)$  with vertex set  $V = \{1, 2, \dots, n\}$  is defined by

$$\vartheta(\Gamma) = \max \{ \langle X, J_n \rangle : X \succeq 0, \langle X, I_n \rangle = 1, X_{ij} = 0 \text{ for all } (i, j) \in E \} \quad (9.20)$$

where  $I_n$  and  $J_n$  denote respectively the identity and the all-one matrices of size  $n$ .

The dual program gives another expression of  $\vartheta(\Gamma)$  (there is no *duality gap* here because  $X = I_n$  is a strictly feasible solution of (9.20) so the Slater condition is fulfilled):

$$\vartheta(\Gamma) = \min \{ t : X \succeq 0, X_{ii} = t - 1, X_{ij} = -1 \text{ for all } (i, j) \notin E \}. \quad (9.21)$$

We have already mentioned that this number provides an upper bound for the independence number  $\alpha(\Gamma)$  of the graph  $\Gamma$ . It also provides a lower bound for the *chromatic number*  $\chi(\bar{\Gamma})$  of the complementary graph  $\bar{\Gamma}$  (the chromatic number of a graph is the minimal number of colors needed to color its vertices so that two connected vertices receive different colors); this is the content of the celebrated *Sandwich theorem* of Lovász [67]:

**Theorem 9.10.**

$$\alpha(\Gamma) \leq \vartheta(\Gamma) \leq \chi(\bar{\Gamma}) \quad (9.22)$$

By modifying the definition of the theta number we get the strengthening  $\alpha(\Gamma) \leq \vartheta'(\Gamma) \leq \vartheta(\Gamma)$ , where

$$\begin{aligned} \vartheta'(\Gamma) = \max & \{ \langle X, J_n \rangle : X \succeq 0, \\ & \langle X, I_n \rangle = 1, X_{ij} = 0 \text{ for all } (i, j) \in E \}. \end{aligned} \quad (9.23)$$

Again, this program has an equivalent dual form:

$$\vartheta'(\Gamma) = \min \{t : X \geq 0, X_{ii} \leq t - 1, X_{ij} \leq -1 \text{ for all } (i, j) \notin E\}. \quad (9.24)$$

### 9.6.2 Generalizations of Lovász $\vartheta$ to the Sphere

In order to obtain the wanted generalization, it is natural to replace in (9.23) the cone of positive semidefinite matrices indexed by the vertex set  $V$  of the underlying graph, with the cone of continuous, positive definite functions on the sphere, defined in Sect. 9.3.1. In fact, there is a small difficulty here due to the fact that, in contrast with a finite dimensional Euclidean space, the space  $C(X)$  of continuous functions on an infinite compact space cannot be identified with its topological dual. Indeed, the *topological dual*  $C(X)^*$  of  $C(X)$  (i.e. the space of continuous linear forms on  $C(X)$ ) is the space  $\mathcal{M}(X)$  of complex valued Borel regular measures on  $X$  (see e.g. Rudin [81, Theorem 6.19], Conway [24, Appendix C]). Consequently, the two forms of  $\vartheta'(\Gamma)$  given by (9.23) and (9.24) lead in the infinite case to two pairs of conic linear programs. As we shall see, the right choice between the two in order to obtain an appropriate bound for  $A(S^{n-1}, \mathcal{Q}, \lambda)$  depends on the set  $\mathcal{Q}$ .

**Definition 9.6.** Let  $\mathcal{Q}^c = \{(x, y) : (x, y) \notin \mathcal{Q} \text{ and } x \neq y\}$ . Let

$$\begin{aligned} \vartheta_1(S^{n-1}, \mathcal{Q}) &= \inf \{t : F \geq 0, F(x, x) \leq t - 1, \\ &\quad F(x, y) \leq -1 \text{ for all } (x, y) \in \mathcal{Q}^c\}, \end{aligned} \quad (9.25)$$

and

$$\begin{aligned} \vartheta_2(S^{n-1}, \mathcal{Q}) &= \sup \{\langle F, 1 \rangle : F \geq 0, F \geq 0, \\ &\quad \langle F, \mathbf{1}_\Delta \rangle = 1, \\ &\quad F(x, y) = 0 \text{ for all } (x, y) \in \mathcal{Q}\}. \end{aligned} \quad (9.26)$$

In the above programs,  $F$  belongs to  $C((S^{n-1})^2)$ . The notation  $F \geq 0$  stands for “ $F$  takes nonnegative values”. The function taking the constant value 1 is denoted  $1$ , so that  $\langle F, 1 \rangle$  equals the integral of  $F$  over  $(S^{n-1})^2$ . In contrast, with  $\langle F, \mathbf{1}_\Delta \rangle$  we mean the integral of the one variable function  $F(x, x)$  over  $\Delta = \{(x, x) : x \in S^{n-1}\}$ . Let us notice that, since  $F$  is continuous in both programs, the sets  $\mathcal{Q}$  and  $\mathcal{Q}^c$  can be replaced by their topological closures  $\overline{\mathcal{Q}}$  and  $\overline{\mathcal{Q}^c}$ .

A *positive semidefinite measure* on  $(S^{n-1})^2$  is one that satisfies  $\langle \lambda, f \rangle \geq 0$  for all  $f \geq 0$ , where

$$\langle \lambda, f \rangle = \int_X f(x) d\lambda(x).$$

and this property is denoted  $\lambda \geq 0$ . In a similar way a nonnegative measure  $\lambda$  is denoted  $\lambda \geq 0$ .

**Theorem 9.11.** *With the above notations and definitions, we have:*

1. *If  $\overline{\mathcal{Q}^c} \cap \mathcal{A} = \emptyset$ , then the program dual to  $\vartheta_1$  reads*

$$\begin{aligned}\vartheta_1^*(S^{n-1}, \mathcal{Q}) &= \sup \{ \langle \lambda, 1 \rangle : \lambda \geq 0, \lambda \geq 0, \\ \lambda(\mathcal{A}) &= 1, \\ \text{supp}(\lambda) &\subset \overline{\mathcal{Q}^c} \cup \mathcal{A} \}\end{aligned}$$

and

$$A(S^{n-1}, \mathcal{Q}, \mu_c) \leq \vartheta_1(S^{n-1}, \mathcal{Q}) = \vartheta_1^*(S^{n-1}, \mathcal{Q}). \quad (9.27)$$

2. *If  $\overline{\mathcal{Q}} \cap \mathcal{A} = \emptyset$ , then the program dual to  $\vartheta_2$  reads*

$$\begin{aligned}\vartheta_2^*(S^{n-1}, \mathcal{Q}) &= \inf \{ t : \lambda \geq 0, \\ \lambda \leq t\mu_{\mathcal{A}} - \mu^2 &\text{ over } (S^{n-1})^2 \setminus \overline{\mathcal{Q}} \},\end{aligned}$$

and

$$A(S^{n-1}, \mathcal{Q}, \mu) \leq \vartheta_2(S^{n-1}, \mathcal{Q}) = \vartheta_2^*(S^{n-1}, \mathcal{Q}). \quad (9.28)$$

*Proof.* The dual programs are computed in a standard way (see Duffin [34], Barvinok [14]). In order to prove that there is no duality gap, one can apply the criterion [14, Theorem 7.2].

For the inequality (9.27), let  $C$  be a maximal spherical code avoiding  $\mathcal{Q}$ . Then, the measure  $\lambda = \delta_{C^2}/|C|$ , where  $\delta$  denotes the Dirac measure, is a feasible solution of  $\vartheta_1^*$  with optimal value  $|C| = A(S^{n-1}, \mathcal{Q}, \mu_c)$ .

For the inequality (9.28), let  $(t, \lambda)$  be a feasible solution of  $\vartheta_2^*$ . Then, if  $C$  avoids  $\mathcal{Q}$ ,  $C^2 \subset (S^{n-1})^2 \setminus \overline{\mathcal{Q}}$ . Thus,  $0 \leq \lambda(C^2) \leq t\mu(C) - \mu(C)^2$ , leading to the wanted inequality  $\mu(C) \leq t$ .  $\square$

*Example 9.1.* In the situations (i) and (ii) above, the set  $\mathcal{Q}$  fulfills the condition 1. of Theorem 9.11 while for (iii) we are in the case 2.

### 9.6.3 Positive Definite Functions Invariant Under the Full Orthogonal Group

It is a classical result of Schoenberg [82] that these functions, in the variables  $(x, y) \in (S^{n-1})^2$ , are exactly the nonnegative linear combinations of *Gegenbauer polynomials* (Sect. 9.3.4) evaluated at the inner product  $x \cdot y$ . We briefly review this classical result, following the lines of Sect. 9.3.

The space  $\text{Hom}_k^n$  of polynomials in  $n$  variables  $x_1, \dots, x_n$  which are homogeneous of degree  $k$  affords an action of the group  $O_n(\mathbb{R})$  acting linearly on the  $n$  variables.

The Laplace operator  $\Delta = \sum_{i=1}^n \frac{\partial^2}{\partial x_i^2}$  commutes with this action, thus the subspace  $\text{Harm}_k^n$  defined by:

$$\text{Harm}_k^n = \{P \in \text{Hom}_k^n : \Delta P = 0\}$$

is a representation of  $O_n(\mathbb{R})$  which turns out to be irreducible (see e.g. Andrews, Askey, Roy [1]). Going from polynomials to polynomial functions on the sphere, we introduce  $H_k^n$ , the subspace of  $C(S^{n-1})$  arising from elements of  $\text{Harm}_k^n$ . Then, for  $d \geq k$ ,  $H_k^n$  is a subspace of the space  $V_d = \text{Pol}_{\leq d}(S^{n-1})$  of polynomial functions on  $S^{n-1}$  of degree up to  $d$ . We have:

**Theorem 9.12.** *The irreducible  $O_n(\mathbb{R})$ -decomposition of  $\text{Pol}_{\leq d}(S^{n-1})$  is given by:*

$$\text{Pol}_{\leq d}(S^{n-1}) = H_0^n \perp H_1^n \perp \cdots \perp H_d^n, \quad (9.29)$$

where, for all  $k \geq 0$ ,  $H_k^n \simeq \text{Harm}_k^n$  is of dimension  $h_k^n = \binom{n+k-1}{k} - \binom{n+k-3}{k-2}$ .

The function  $Y_k(d_\theta(x, y))$  associated to  $H_k^n$  following (9.14) equals:

$$Y_k(d_\theta(x, y)) = h_k^n P_k^{((n-3)/2, (n-3)/2)}(x \cdot y). \quad (9.30)$$

*Proof.* For the sake of completeness we sketch a proof. Because  $S^{n-1}$  is two-point homogeneous under  $O_n(\mathbb{R})$ , the algebra  $(V_d)^{(2)}$  related to the finite dimensional  $O_n(\mathbb{R})$ -space  $V_d$  is equal to the set of polynomial functions in the variable  $t = x \cdot y$  of degree up to  $d$ , thus has dimension  $d+1$ . On the other hand, it can be shown that the  $d+1$  subspaces  $H_k^n$ , for  $0 \leq k \leq d$ , are non zero and pairwise distinct. Then, the statement (9.29) follows from the equality of the dimensions of  $(V_d)^{(2)}$  and of  $\text{End}_{O_n(\mathbb{R})}(V_d)$  which are isomorphic algebras (9.12). It is worth to notice that we end up with a proof that the spaces  $H_k^n$  are irreducible, without referring to the irreducibility of the spaces  $\text{Harm}_k^n$  (which can be proved in a similar way). The formula for  $h_k^n$  follows by induction.

The functions  $Y_k(d_\theta(x, y))$  associated to the decomposition (9.29) must be polynomials in the variable  $x \cdot y$  of degree  $k$ ; let us denote them temporary by  $Q_k(t)$ . A change of variables shows that, for a function  $f(x) \in C(S^{n-1})$  of the form  $f(x) = \varphi(x \cdot y)$  for some  $y \in S^{n-1}$ ,

$$\int_{S^{n-1}} f(x) d\mu(x) = c_n \int_{-1}^1 \varphi(t) (1-t^2)^{(n-3)/2} dt$$

for some constant  $c_n$ . Then, because the subspaces  $H_k^n$  are pairwise orthogonal, the polynomials  $Q_k$  must satisfy the orthogonality conditions

$$\int_{-1}^1 Q_k(t) Q_l(t) (1-t^2)^{(n-3)/2} dt = 0$$

for all  $k \neq l$  thus they must be equal to the Gegenbauer polynomials up to a multiplicative factor. Integrating over  $S^{n-1}$  the formula (9.9) when  $x = y$  shows that  $Y_k(0) = h_k^n$  thus computes this factor.  $\square$

**Corollary 9.2.** *Let  $P_k^n(t) = P_k^{((n-3)/2,(n-3)/2)}(t)$ . Then,  $F \in C(S^{n-1})_{\geq 0}^{O_n(\mathbb{R})}$  if and only if*

$$F(x, y) = \sum_{k \geq 0} f_k P_k^n(x \cdot y), \text{ with } f_k \geq 0 \text{ for all } k \geq 0. \quad (9.31)$$

#### 9.6.4 Positive Definite Functions Invariant Under the Stabilizer of One Point

We fix an element  $e \in S^{n-1}$  and let  $G = \text{Stab}(e, O_n(\mathbb{R}))$ . Then, the orbit  $O(x)$  of  $x \in S^{n-1}$  under  $G$  is the set

$$O(x) = \{y \in S^{n-1} : e \cdot y = e \cdot x\}.$$

The orbit  $O((x, y))$  of  $(x, y) \in S^{n-1}$  equals

$$O((x, y)) = \{(z, t) \in (S^{n-1})^2 : (e \cdot z, e \cdot t, z \cdot t) = (e \cdot x, e \cdot y, x \cdot y)\}.$$

In other words, the orbits of  $G$  on  $(S^{n-1})^2$  are parametrized by the triple of real numbers  $(e \cdot x, e \cdot y, x \cdot y)$ . So the  $G$ -invariant positive definite functions on  $S^{n-1}$  are functions of the three variables  $u = e \cdot x$ ,  $v = e \cdot y$ ,  $t = x \cdot y$ . Their expression is computed in Bachoc and Vallentin [5].

**Theorem 9.13.** *The irreducible decomposition of  $V_d = \text{Pol}_{\leq d}(S^{n-1})$  under the action of  $G = \text{Stab}(e, S^{n-1}) \simeq O_{n-1}(\mathbb{R})$  is given by:*

$$\text{Pol}_{\leq d}(S^{n-1}) \simeq \bigoplus_{k=0}^d \left( \text{Harm}_k^{n-1} \right)^{d-k+1}. \quad (9.32)$$

The coefficients of the matrix  $Y_k = Y_k^n$  of size  $d - k + 1$  associated to  $\text{Harm}_k^{n-1}$  are equal to:

$$Y_{k,ij}^n(u, v, t) = u^i v^j Q_k^{n-1}(u, v, t) \quad (9.33)$$

where  $0 \leq i, j \leq d - k$ , and

$$Q_k^{n-1}(u, v, t) = \left( (1 - u^2)(1 - v^2) \right)^{k/2} P_k^{n-1} \left( \frac{t - uv}{\sqrt{(1 - u^2)(1 - v^2)}} \right)$$

*Proof.* We refer to Bachoc and Vallentin [5] for the details. In order to obtain the  $G$ -decomposition of the space  $\text{Pol}_{\leq d}(S^{n-1})$  we can start from the  $O_n(\mathbb{R})$ -decomposition (9.29). The  $O_n(\mathbb{R})$ -irreducible subspaces  $H_k^n$  split into smaller subspaces following the  $O_{n-1}(\mathbb{R})$ -isomorphisms (see e.g. Vilenkin and Klimyk [96]):

$$\text{Harm}_k^n \simeq \bigoplus_{i=0}^k \text{Harm}_i^{n-1} \quad (9.34)$$

leading to (9.32). The expression (9.33) follows from the definition (9.9) of  $E_{k,ij}(x,y)$  and from the construction of convenient basis  $(e_{k,i,s})_s$ . These basis arise from explicit embeddings of  $H_k^{n-1}$  into  $V_d$ , defined as follows. For  $x \in S^{n-1}$ , let  $x = ue + \sqrt{1-u^2}\zeta$  where  $\zeta \in (\mathbb{R}e)^\perp$  and  $\zeta \cdot \zeta = 1$ . Let  $\varphi_{k,i}$  send  $f \in H_k^{n-1}$  to  $\varphi_{k,i}(f)$  where

$$\varphi_{k,i}(f)(x) = u^i(1-u^2)^{k/2}f(\zeta).$$

Then,  $\{e_{k,i,s} : 1 \leq s \leq h_k^{n-1}\}$  is taken to be the image by  $\varphi_{k,i}$  of an orthonormal basis of  $H_k^{n-1}$ .  $\square$

### 9.6.5 Reduction Using Symmetries

If the set  $\mathcal{Q}$  is invariant under a subgroup  $G$  of  $O_n(\mathbb{R})$ , then the feasible sets of the conic linear programs (9.25) and (9.26) can be restricted to the  $G$ -invariant positive definite functions  $F$ . Indeed, symmetry reduction of finite dimensional semidefinite programs extends to infinite compact spaces, with the now familiar trick that replaces the finite average over the elements of a finite group by the integral for the Haar measure of the group. We apply this general principle to the programs (9.25) and (9.26) and we focus on the cases (i)–(iii) above.

#### 9.6.5.1 Case (i): $\mathcal{Q} = \{(x,y) : d_\theta(x,y) \in ]0, \theta_{\min}[ \}$ .

The set  $\mathcal{Q}$  is invariant under  $O_n(\mathbb{R})$ . According to Theorem 9.11 we consider the program (9.25) where  $F \geq 0$  can be replaced by (9.31). The program  $\vartheta_1$  becomes after a few simplifications:

$$\vartheta_1(S^{n-1}, \mathcal{Q}) = \inf \left\{ 1 + \sum_{k \geq 1} f_k : f_k \geq 0, 1 + \sum_{k \geq 1} f_k P_k^n(t) \leq 0 \quad t \in [-1, s] \right\}, \quad (9.35)$$

where  $s = \cos(\theta_{\min})$ . One recognises in (9.35) the so-called *Delsarte linear programming bound* for the maximal number of elements of a spherical code with minimal angular distance  $\theta_{\min}$ , see Delsarte et al. [32], Kabatiansky and Levenshtein [50], Conway and Sloane [25, Chap. 9]. The optimal value of the linear program (9.35) is not known in general, although explicit feasible solutions leading to very

good bounds and also to asymptotic bounds have been constructed (Kabatiansky and Levenshtein [50], Levenshtein [65], Odlyzko and Sloane [75]). Moreover, this infinite dimensional linear program can be efficiently approximated by semidefinite programs defined in the following way: in order to deal with only a finite number of variables  $f_k$ , one restricts to  $k \leq d$  (it amounts to restrict to  $G$ -invariant positive definite functions of  $V_d^{(2)}$ ). Then the polynomial  $1 + \sum_{k=1}^d f_k P_k^n$  is required in (9.35) to be nonpositive over a certain interval of real numbers. By the theorem of Lukács concerning non-negative polynomials (see e.g. Szegö [89, Chap. 1.21]), this can be expressed as a sums of square condition, hence as a semidefinite program. Then, when  $d$  varies, one obtains a sequence of semidefinite programs approaching  $\vartheta_1(S^{n-1}, \Omega)$  from above.

#### 9.6.5.2 Case (ii): $\Omega = \{(x, y) : d_\theta(x, y) = \theta\}$ .

The set  $\Omega$  is again invariant under  $O_n(\mathbb{R})$  but now we deal with (9.26), which becomes:

$$\vartheta_2(S^{n-1}, \Omega) = \sup \left\{ f_0 : f_k \geq 0, \sum_{k \geq 0} f_k = 1, \quad \sum_{k \geq 0} f_k P_k^n(s) = 0 \right\} \quad (9.36)$$

where  $s = \cos(\theta)$ . This linear program has infinitely many variables but only two constraints. Its optimal value turns to be easy to determine (we refer to Bachoc et al. [10] for a proof).

**Theorem 9.14.** *Let  $m(s)$  be the minimum of  $P_k^n(s)$  for  $k = 0, 1, 2, \dots$ . Then*

$$\vartheta_2(S^{n-1}, \Omega) = \frac{m(s)}{m(s) - 1}.$$

#### 9.6.5.3 Case (iii): $\Omega = \{(x, y) : d_\theta(x, y) \in ]0, \theta_{\min}[ \text{ or } (x, y) \notin \text{Cap}(e, \phi)^2\}$ .

This set is invariant by the smaller group  $G = \text{Stab}(e, O_n(\mathbb{R}))$ . Like in case (i), the program  $\vartheta_1$  must be considered and in this program  $F$  can be assumed to be  $G$ -invariant.

From Stone–Weierstrass theorem (see e.g. Conway [24, Theorem 8.1]), the elements of  $C((S^{n-1})^2)$  can be uniformly approximated by those of  $V_d^{(2)}$ . In addition, one can prove that the elements of  $C((S^{n-1})^2)_{\geq 0}$  can be uniformly approximated by positive definite functions belonging to  $V_d^{(2)}$ . We introduce:

$$\begin{aligned} \vartheta_1^{(d)}(S^{n-1}, \Omega) &= \inf \left\{ t : F \in \left(V_d^{(2)}\right)_{\geq 0}, F(x, x) \leq t - 1, \right. \\ &\quad \left. F(x, y) \leq -1 \text{ for all } (x, y) \in \Omega^c \right\}. \end{aligned} \quad (9.37)$$

So,  $\vartheta_1(S^{n-1}, \mathcal{Q}) \leq \vartheta_1^{(d)}(S^{n-1}, \mathcal{Q})$ , and the limiting value of  $\vartheta_1^{(d)}(S^{n-1}, \mathcal{Q})$  when  $d$  goes to infinity equals  $\vartheta_1(S^{n-1}, \mathcal{Q})$ . Since  $\mathcal{Q}$  and  $V_d$  are invariant by  $G$ , one can moreover assume that  $F$  is  $G$ -invariant. From Theorems 9.7 and 9.13, we have an expression for  $F$ :

$$F(x, y) = \sum_{k=0}^d \langle F_k, Y_k(u, v, t) \rangle, \quad F_k \geq 0,$$

where the matrices  $F_k$  are of size  $d - k + 1$ . Replacing in  $\vartheta_1^{(d)}$  leads to:

$$\begin{aligned} \vartheta_1^{(d)}(S^{n-1}, \mathcal{Q}) &= \inf \left\{ t : F_k \geq 0, \sum_{k=0}^d \langle F_k, Y_k(u, u, 1) \rangle \leq t \quad u \in [s', 1], \right. \\ &\quad \left. \sum_{k=0}^d \langle F_k, Y_k(u, v, t) \rangle \leq -1 \quad s' \leq u \leq v \leq 1 \quad -1 \leq t \leq s \right\} \end{aligned}$$

with  $s = \cos(\theta)$  and  $s' = \cos(\phi)$ . The left hand sides of the inequalities are polynomials in three variables. Again, these constraints can be relaxed using sums of squares in order to boil down to true semidefinite programs. We refer to Bachoc and Vallentin [6] for the details, and for numerical computations of upper bounds for codes in spherical caps with given minimal angular distance.

### 9.6.6 Further Applications

In Bachoc and Vallentin [5] it is shown how Delsarte linear programming bound [28] can be improved with semidefinite constraints arising from the matrices  $Y_k^n$  (9.33). The idea is very much the same as for the Hamming space given in Schrijver [84] and explained in Sect. 9.4 : instead of considering constraints on pairs of points only, one exploits constraints on triples of points. More precisely, if  $S_k^n(u, v, t)$  denotes the symmetrization of  $Y_k^n(u, v, t)$  in the variables  $(u, v, t)$ , then the following semidefinite property holds for all spherical code  $C$ :

$$\sum_{(x,y,z) \in C^3} S_k^n(x \cdot y, y \cdot z, z \cdot x) \geq 0. \quad (9.38)$$

From (9.38), it is possible to define a semidefinite program whose optimal value upper bounds the number of elements of a code with given minimal angular distance. In Bachoc and Vallentin [5], Mittelmann and Vallentin [71], new upper bounds for the kissing number have been obtained for the dimensions  $n \leq 24$  with this method. We give next a simplified version of the semidefinite program used in [5]. Another useful version is given in Bachoc and Vallentin [8] for proving that the maximal angular distance of 10 points on  $S^3$  is  $\cos(1/6)$ .

**Theorem 9.15.** *The optimal value of the semidefinite program:*

$$\inf \left\{ 1 + \langle F_0, J_{d+1} \rangle : F_k \geq 0 \sum_{k=0}^d \langle F_k, S_k^n(u, u, 1) \rangle \leq -\frac{1}{3}, \quad -1 \leq u \leq s \right. \\ \left. \sum_{k=0}^d \langle F_k, S_k^n(u, v, t) \rangle \leq 0, \quad -1 \leq u, v, t \leq s \right\} \quad (9.39)$$

is an upper bound for the number  $A(S^{n-1}, \Omega, \mu_c)$  where  $\Omega = \{(x, y) \in (S^{n-1})^2 : s < x \cdot y < 1\}$ , i.e. for the maximal number of elements of a spherical code with minimal angular distance at least equal to  $\theta_{\min} = \arccos(s)$ .

*Proof.* Let  $(F_0, \dots, F_k)$  a feasible solution of (9.39). Let

$$F(x, y, z) = \sum_{k=0}^d \langle F_k, S_k^n(x \cdot y, y \cdot z, z \cdot x) \rangle.$$

If  $C$  is a spherical code, we consider  $\Sigma = \sum_{(x,y,z) \in C^3} F(x, y, z)$ . We have:

$$0 \leq \Sigma = \sum_{x \in C} F(x, x, x) + \sum_{\|x, y, z\|=2} F(x, x, y) + \sum_{\|x, y, z\|=3} F(x, y, z)$$

where the inequality holds because of (9.38). Then, taking  $S_0^n(1, 1, 1) = J_{d+1}$  and  $S_k^n(1, 1, 1) = 0$  for  $k \geq 1$  into account, we have  $F(x, x, x) = \langle F_0, J_{d+1} \rangle$ . If moreover  $d_\theta(C) \geq \theta_{\min}$ , we can apply the constraint inequalities of the program to the second and third terms of the right hand side. We obtain:

$$0 \leq \Sigma \leq \langle F_0, J_{d+1} \rangle |C| - |C|(|C| - 1)$$

leading to the inequality  $|C| \leq 1 + \langle F_0, J_{d+1} \rangle$ . □

## 9.7 Sums of Squares

A fundamental task in polynomial optimization and in real algebraic geometry is to decide and certify whether a polynomial with real coefficients in  $n$  indeterminates can be written as a *sum of squares*: Given  $p \in \mathbb{R}[x_1, \dots, x_n]$  do there exist polynomials  $q_1, \dots, q_m \in \mathbb{R}[x_1, \dots, x_n]$  so that

$$p = q_1^2 + q_2^2 + \dots + q_m^2 ?$$

This problem can be reformulated as a semidefinite feasibility problem: Let  $z$  be a vector containing a basis of  $\mathbb{R}[x_1, \dots, x_n]_{\leq d}$  the space of polynomials of degree at most  $d$ . For example, let  $z$  be the vector containing the monomial basis

$$z = (1, x_1, x_2, \dots, x_n, x_1^2, x_1 x_2, x_2^2, \dots, x_n^d)$$

which has length  $\binom{n+d}{d}$ . A polynomial  $p \in \mathbb{R}[x_1, \dots, x_n]$  of degree  $2d$  is a sum of square if and only if there is a positive semidefinite matrix  $X$  of size  $\binom{n+d}{d} \times \binom{n+d}{d}$  so that the  $\binom{n+2d}{2d}$  linear – linear in the entries of  $X$  – equations

$$p(x_1, \dots, x_n) = z^T X z$$

hold.

This semidefinite feasibility problem can be simplified if the polynomial  $p$  has symmetries. The method has been worked out by Gatermann and Parrilo in [37]. In this section we give the main ideas of the method. For details and further we refer to the original article.

### 9.7.1 Basics from Invariant Theory

We start by explaining what we mean that a polynomial has symmetries. Again to simplify the presentation of the theory we consider the complex case only.

Let  $G$  be a finite group acting on  $\mathbb{C}^n$ . This group action induces a group action on the polynomial ring  $\mathbb{C}[x_1, \dots, x_n]$  by

$$(gp)(x_1, \dots, x_n) = p(g^{-1}(x_1, \dots, x_n)),$$

and we say that a polynomial  $p$  is  $G$ -invariant if  $gp = p$  for all  $g \in G$ . The set  $\mathbb{C}[x_1, \dots, x_n]^G$  of all  $G$ -invariant polynomials is a ring, the *invariant ring*. By Hilbert's finiteness theorem it is generated by finitely many  $G$ -invariant polynomials. Even more is true: Since the invariant ring has the Cohen–Macaulay property it admits a Hironaka decomposition: There are  $G$ -invariant polynomials  $\eta_i, \theta_j \in \mathbb{C}[x_1, \dots, x_n]$  so that

$$\mathbb{C}[x_1, \dots, x_n]^G = \bigoplus_{i=1}^r \eta_i \mathbb{C}[\theta_1, \dots, \theta_s], \quad (9.40)$$

hence, every invariant polynomial can be *uniquely* written as a polynomial in the polynomials  $\eta_i$  and  $\theta_j$  where  $\eta_i$  only occurs linearly. We refer to Sturmfels [88, Chap. 2.3] for the definitions and proofs; we only need the existence of a Hironaka decomposition here.

### 9.7.2 Sums of Squares with Symmetries

We consider the action of the finite group  $G$  restricted to the  $\binom{n+d}{d}$ -dimensional vector space of complex polynomials of degree at most  $d$ . This defines a unitary representation

$$\pi : G \rightarrow \mathrm{Gl}(\mathbb{C}[x_1, \dots, x_n]_{\leq d}).$$

From now on, by using the monomial basis, we see  $\pi(g)$  as a regular matrix in  $\mathbb{C}^{(\frac{n+d}{d}) \times (\frac{n+d}{d})}$ .

*Example 9.2.* For instance, the matrix  $g^{-1} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$  acts on  $\mathbb{C}^2$  and so on the polynomial  $p = 1 + x_1 + x_2 + x_1^2 + x_1x_2 + x_2^2 \in \mathbb{C}[x_1, x_2]_{\leq 2}$  by

$$\begin{aligned} (gp)(x_1, x_2) &= p(x_1 + 2x_2, 3x_1 + 4x_2) \\ &= 1 + (x_1 + 2x_2) + (3x_1 + 4x_2) + (x_1 + 2x_2)^2 \\ &\quad + (x_1 + 2x_2)(3x_1 + 4x_2) + (3x_1 + 4x_2)^2 \\ &= 1 + (x_1 + 2x_2) + (3x_1 + 4x_2) + (x_1^2 + 4x_1x_2 + 4x_2^2) \\ &\quad + (3x_1^2 + 10x_1x_2 + 8x_2^2) + (9x_1^2 + 24x_1x_2 + 16x_2^2). \end{aligned}$$

and so defines the matrix

$$\pi(g) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 3 & 0 & 0 & 0 \\ 0 & 2 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 3 & 9 \\ 0 & 0 & 0 & 4 & 10 & 24 \\ 0 & 0 & 0 & 4 & 8 & 16 \end{pmatrix}.$$

Let  $p \in \mathbb{R}[x_1, \dots, x_n]$  be a polynomial which is a sum of squares and which is  $G$ -invariant. Thus we have a positive semidefinite matrix  $X \in \mathbb{R}^{(\frac{n+d}{d}) \times (\frac{n+d}{d})}$  so that

$$p(x_1, \dots, x_n) = z^\top X z = z^* X z,$$

and for every  $g \in G$  we have

$$gp(x_1, \dots, x_n) = (\pi(g)^* z)^* X (\pi(g)^* z) = z^* \pi(g) X \pi(g)^* z.$$

Hence,  $X$  is  $G$ -invariant and lies in  $\left(\mathbb{C}^{(\frac{n+d}{d}) \times (\frac{n+d}{d})}\right)^G$ , the commutant algebra of the matrix  $*$ -algebra spanned by the matrices  $\pi(g)$  with  $g \in G$ . So by Theorem 9.5 there are numbers  $D, m_1, \dots, m_D$  and a  $*$ -isomorphism

$$\varphi : \left(\mathbb{C}^{(\frac{n+d}{d}) \times (\frac{n+d}{d})}\right)^G \rightarrow \bigoplus_{k=1}^D \mathbb{C}^{m_k \times m_k}.$$

Hence, cf. Step 2 (second version) in Sect. 9.1.2, we can write the polynomial  $p$  in the form

$$p(x_1, \dots, x_n) = z^* \left( \sum_{k=1}^D \sum_{u,v=1}^{m_k} x_{k,uv} \varphi^{-1}(E_{k,uv}) \right) z$$

with  $D$  positive semidefinite matrices

$$X_k = (x_{k,uv})_{1 \leq u,v \leq m_k}, \quad k = 1, \dots, D.$$

We define  $D$  matrices  $E_1, \dots, E_D \in (\mathbb{C}[x_1, \dots, x_n]^G)^{m_k \times m_k}$  with  $G$ -invariant polynomial entries by

$$(E_k)_{uv} = \left( \varphi^{-1}(E_{k,uv}) \right)_{\prod_i x_i^{\alpha_i}, \prod_i x_i^{\beta_i}} \prod_i x_i^{\alpha_i + \beta_i},$$

where we consider matrices in  $\mathbb{C}^{(\binom{n+d}{d}) \times (\binom{n+d}{d})}$  as matrices whose rows and columns are indexed by monomials  $\prod_i x_i^{\alpha_i}$ . Then, the polynomial  $p$  has a representation of the form

$$p(x_1, \dots, x_n) = \sum_{k=1}^D \langle X_k, E_k \rangle.$$

Since the entries of  $E_k$  are  $G$ -invariant polynomials we can use a Hironaka decomposition to represent them in terms of the invariants  $\eta_i$  and  $\theta_j$ . We summarize our discussion in the following theorem.

**Theorem 9.16.** *Let  $p$  be a  $G$ -invariant polynomial of degree  $2d$  which is a sum of squares. Then there are numbers  $D, m_1, \dots, m_D$  so that  $p$  has a representation of the form*

$$p(x_1, \dots, x_n) = \sum_{k=1}^D \langle X_k, E_k \rangle,$$

where  $X_k \in \mathbb{C}^{m_k \times m_k}$  are positive semidefinite Hermitian matrices, and where

$$E_k \in \left( \bigoplus_{i=1}^r \eta_i \mathbb{C}[\theta_1, \dots, \theta_s] \right)^{m_k \times m_k}$$

are matrices whose entries are polynomials (determined by a Hironaka decomposition of  $\mathbb{C}[x_1, \dots, x_n]^G$  and by the  $*$ -isomorphism  $\varphi$ ).

## 9.8 More Applications

In the last years many results were obtained for semidefinite programs which are symmetric. This was done for a variety of problems and applications. In this final section we want to give a brief, and definitely not complete, guide to the extensive and growing literature.

### 9.8.1 *Interior Point Algorithms*

Kanno et al. [51] consider structural properties of search directions in primal-dual interior-point methods for solving invariant semidefinite programs and apply this to truss optimization problems. de Klerk and Pasechnik [56] show how the dual scaling method can be implemented to exploit the particular data structure where the data matrices come from a low-dimensional matrix algebra.

### 9.8.2 *Combinatorial Optimization*

Using symmetry in semidefinite programs has been used in combinatorial optimization for a variety of problems: quadratic assignment problem (de Klerk and Sotirov [60]), travelling salesman problem (de Klerk et al. [59]), graph coloring (Gvozdenović and Laurent [44], [45], Gvozdenović [43]), Lovász theta number (de Klerk et al. [55]).

### 9.8.3 *Polynomial Optimization*

Jansson et al. [49] work out how constrained polynomial optimization problems behave which are invariant under the action of the symmetric group or the cyclic group. Among many other things, Cimprič et al. [23] extend the discussion of Gatermann and Parrilo [37] from finite groups to compact groups. Cimprič [22] transfers the method to compute minima of the spectra of differential operators. In [17] Bosse constructs symmetric polynomials which are non-negative but not sums of squares.

### 9.8.4 *Low Distortion Geometric Embedding Problems*

Linial et al. [66] give lower bounds for low distortion embedding of graphs into Euclidean space depending on the girth. Vallentin [93] finds explicit optimal low distortion embeddings for several families of distance regular graphs. Both papers construct feasible solutions of semidefinite programs by symmetry reduction and by using the theory of orthogonal polynomials.

### 9.8.5 *Miscellaneous*

Bai et al. [11] exploit symmetry in truss topology optimization and Boyd et al. [18] in the analysis of fast mixing Markov chains on graphs.

### 9.8.6 Software

Pasechnik and Kini [78] develop a software package for the computer algebra system GAP for computing with the regular  $*$ -representation for matrix  $*$ -algebras coming from coherent configurations.

### 9.8.7 Surveys and Lecture Notes

Several surveys and lecture notes on symmetry in semidefinite programs with different aims were written in the last years. The lecture notes of Bachoc [4] especially discuss applications in coding theory and extend those of Vallentin [95] which focuses on aspects from harmonic analysis. The survey [53] of de Klerk discusses next to symmetry also the exploitation of other structural properties of semidefinite programs like low rank or sparsity.

**Acknowledgements** We thank the referee for the helpful suggestions and comments. The fourth author was supported by Vidi grant 639.032.917 from the Netherlands Organization for Scientific Research (NWO).

## References

1. Andrews, G.E., Askey, R., Roy, R.: Special functions. Cambridge University Press, Cambridge (1999)
2. Astola, J.: The Lee-scheme and bounds for Lee-codes. *Cybernet. Systems* **13**, 331–343 (1982)
3. Bachoc, C.: Linear programming bounds for codes in Grassmannian spaces. *IEEE Trans. Inf. Th.* **52**, 2111–2125 (2006)
4. Bachoc, C.: Semidefinite programming, harmonic analysis and coding theory. arXiv:0909.4767 [cs.IT] (2009)
5. Bachoc, C., Vallentin, F.: New upper bounds for kissing numbers from semidefinite programming. *J. Amer. Math. Soc.* **21**, 909–924 (2008)
6. Bachoc, C., Vallentin, F.: Semidefinite programming, multivariate orthogonal polynomials, and codes in spherical caps, special issue in the honor of Eiichi Bannai, *Europ. J. Comb.* **30** 625–637 (2009)
7. Bachoc, C., Vallentin, F.: More semidefinite programming bounds (extended abstract). pages 129–132 in Proceedings “DMHF 2007: COE Conference on the Development of Dynamic Mathematics with High Functionality”, October 2007, Fukuoka, Japan. (2007)
8. Bachoc, C., Vallentin, F.: Optimality and uniqueness of the (4,10,1/6) spherical code. *J. Comb. Theory Ser. A* **116**, 195–204 (2009)
9. Bachoc, C., Zémor, G.: Bounds for binary codes relative to pseudo-distances of  $k$  points. *Adv. Math. Commun.* **4**, 547–565 (2010)
10. Bachoc, C., Nebe G., de Oliveira Filho, F.M., Vallentin, F.: Lower bounds for measurable chromatic numbers. *Geom. Funct. Anal.* **19**, 645–661 (2009)
11. Bai, Y., de Klerk, E., Pasechnik, D.V., Sotirov, R.: Exploiting group symmetry in truss topology optimization. *Optimization and Engineering* **10**, 331–349 (2009)

12. Bannai, E., Ito, T.: Algebraic combinatorics. I.. The Benjamin/Cummings Publishing Co. Inc., Menlo Park, CA (1984)
13. Barg, A., Purkayastha, P.: Bounds on ordered codes and orthogonal arrays. *Moscow Math. Journal* **9**, 211–243 (2009)
14. Barvinok, A.: A course in convexity. Graduate Studies in Mathematics **54**, American Mathematical Society (2002)
15. Berger, M.: A Panoramic View of Riemannian Geometry. Springer-Verlag (2003)
16. Bochner, S.: Hilbert distances and positive definite functions. *Ann. of Math.* **42**, 647–656 (1941)
17. Bosse, H.: Symmetric, positive polynomials, which are not sums of squares. Series: CWI. Probability, Networks and Algorithms [PNA], Nr. E0706 (2007)
18. Boyd, S., Diaconis, P., Parrilo, P.A., Xiao, L.: Symmetry analysis of reversible Markov chains. *Internet Mathematics* **2**, (2005)
19. Brouwer, A.E., Cohen, A.M., Neumaier, A.: Distance-regular graphs. Springer-Verlag, Berlin (1989)
20. Bump, D.: Lie Groups. Graduate Text in Mathematics **225**, Springer-Verlag (2004)
21. Cameron, P. J.: Coherent configurations, association schemes and permutation groups. pages 55–71 in *Groups, combinatorics & geometry (Durham, 2001)*, World Sci. Publishing, River Edge, NJ (2003)
22. Cimprič, J.: A method for computing lowest eigenvalues of symmetric polynomial differential operators by semidefinite programming. *J. Math. Anal. Appl.* **369**, 443–452 (2010)
23. Cimprič, J., Kuhlmann, S., Scheiderer, C.: Sums of squares and moment problems in equivariant situations. *Trans. Amer. Math. Soc.* **361**, 735–765 (2009).
24. Conway, J.B.: A course in functional analysis. Graduate Text in Mathematics **96**, Springer-Verlag (2007)
25. Conway, J.H., Sloane, N.J.A.: Sphere Packings, Lattices and Groups. third edition, Springer-Verlag, New York (1999)
26. Creignou, J., Diet, H.: Linear programming bounds for unitary codes. *Adv. Math. Commun.* **4**, 323–344 (2010)
27. Davidson, K.R.: C\*-Algebras by Example. Fields Institute Monographs **6**, American Mathematical Society (1996)
28. Delsarte, P.: An algebraic approach to the association schemes of coding theory. *Philips Res. Rep. Suppl.* (1973)
29. Delsarte, P., Goethals, J. M.: Alternating bilinear forms over  $GF(q)$ . *J. Comb. Th. A* **19**, 26–50 (1975)
30. Delsarte, P.: Hahn polynomials, discrete harmonics and  $t$ -designs. *SIAM J. Appl. Math.* **34**, 157–166 (1978)
31. Delsarte, P.: Bilinear forms over a finite field, with applications to coding theory. *J. Comb. Th. A* **25**, 226–241 (1978)
32. Delsarte, P., Goethals, J.M., Seidel, J.J.: Spherical codes and designs. *Geom. Dedicata* **6**, 363–388 (1977)
33. de Klerk, E., Dobre, C., Pasechnik, D.V.: Numerical block diagonalization of matrix \*-algebras with application to semidefinite programming. To appear in *Math. Program.*, Ser. B. (2009)
34. Duffin, R.J.: Infinite Programs. In: H.W. Kuhn, A.W. Tucker (eds.) *Linear inequalities and related systems*, Princeton Univ. Press, 157–170 (1956)
35. Dunkl, C.F.: A Krawtchouk polynomial addition theorem and wreath product of symmetric groups. *Indiana Univ. Math. J.* **25**, 335–358 (1976)
36. Ericson, T., Zinoviev, V.: Codes on Euclidean spheres. North-Holland (2001)
37. Gatermann, K., Parrilo, P.A.: Symmetry groups, semidefinite programs, and sums of squares. *J. Pure Appl. Alg.* **192**, 95–128 (2004)
38. Gijswijt, D.C.: Matrix Algebras and Semidefinite Programming Techniques for Codes. Dissertation, University of Amsterdam (2005)
39. Gijswijt, D.C.: Block diagonalization for algebra's associated with block codes. arXiv:0910.4515 [math.OC] (2009)

40. Gijswijt, D.C., Mittelmann, H.D., Schrijver, A.: Semidefinite code bounds based on quadruple distances. arXiv.math:1005.4959 [math.CO] (2010)
41. Gijswijt, D.C., Schrijver, A., Tanaka, H.: New upper bounds for nonbinary codes based on the Terwilliger algebra and semidefinite programming. *J. Comb. Theory Ser. A* **113**, 1719–1731 (2006)
42. Goemans, M.X., Williamson, D.P.: Approximation algorithms for MAX-3-CUT and other problems via complex semidefinite programming. *J. Comput. System Sci.* **68**, 442–470 (2004)
43. Gvozdenović, N.: Approximating the stability number and the chromatic number of a graph via semidefinite programming. Dissertation, University of Amsterdam (2008)
44. Gvozdenović, N., Laurent, M.: The operator  $\Psi$  for the chromatic number of a graph. *SIAM J. Optim.* **19**, 572–591 (2008)
45. Gvozdenović, N., Laurent, M.: Computing semidefinite programming lower bounds for the (fractional) chromatic number via block-diagonalization. *SIAM J. Optim.* **19**, 592–615 (2008)
46. Gvozdenović, N., Laurent, M., Vallentin, F.: Block-diagonal semidefinite programming hierarchies for 0/1 programming. *Oper. Res. Lett.* **37**, 27–31 (2009)
47. Horn, R.A., Johnson, C.R.: Matrix analysis. Cambridge University Press, Cambridge (1990)
48. James, A.T., Constantine, A.G.: Generalized Jacobi polynomials as spherical functions of the Grassmann manifold. *Proc. London Math. Soc.* **29**, 174–192 (1974)
49. Jansson, L., Lasserre, J.B., Riener, C., Theobald, T.: Exploiting symmetries in SDP-relaxations for polynomial optimization. Optimization Online, September 2006, (2006)
50. Kabatiansky, G.A., Levenshtein, V.I.: Bounds for packings on a sphere and in space. *Problems of Information Transmission* **14**, 1–17 (1978)
51. Kanno, Y., Ohsaki, M., Murota, K., Katoh, N.: Group symmetry in interior-point methods for semidefinite program. *Optimization and Engineering* **2**, 293–320 (2001)
52. Kleitman, D.J.: The crossing number of  $K_{5,n}$ . *J. Comb. Theory Ser. B* **9**, 315–323 (1970)
53. de Klerk, E.: Exploiting special structure in semidefinite programming: A survey of theory and applications. *European Journal of Operational Research* **201**, 1–10 (2010)
54. de Klerk, E., Maharry, J., Pasechnik, D.V., Richter, R.B., Salazar, G.: Improved bounds for the crossing numbers of  $K_{m,n}$  and  $K_n$ . *SIAM J. Disc. Math.* **20**, 189–202 (2006)
55. de Klerk, E., Newman, M.W., Pasechnik, D.V., Sotirov R.: On the Lovasz  $\theta$ -number of almost regular graphs with application to Erdős-Rényi graphs. *European J. Combin.* **31**, 879–888 (2009)
56. de Klerk, E., Pasechnik, D.V.: Solving SDP's in non-commutative algebras part I: the dual-scaling algorithm. Discussion paper from Tilburg University, Center for economic research (2005)
57. de Klerk, E., Pasechnik, D.V.: A note on the stability number of an orthogonality graph. *European J. Combin.* **28**, 1971–1979 (2007)
58. de Klerk, E., Pasechnik, D.V., Schrijver, A.: Reduction of symmetric semidefinite programs using the regular  $*$ -representation. *Math. Program., Ser. B* **109**, 613–624 (2007)
59. de Klerk, E., Pasechnik, D.V., Sotirov, R.: On Semidefinite Programming Relaxations of the Travelling Salesman Problem. Discussion paper from Tilburg University, Center for economic research (2007)
60. de Klerk, E., Sotirov, R.: Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. *Math. Program. Ser. A* **122**, 225–246 (2010)
61. Knuth, D.E.: The sandwich theorem. *Electron. J. Combin.* **1**, (1994)
62. Lasserre, J.B.: An explicit exact SDP relaxation for nonlinear 0/1 programs. In: K. Aardal and A.M.H. Gerards (eds.) *Lecture Notes in Computer Science* **2081** (2001)
63. Laurent, M.: Strengthened semidefinite programming bounds for codes. *Math. Program. Ser. B* **109**, 239–261 (2007)
64. Laurent, M.: Sums of squares, moment matrices and optimization over polynomials. In: M. Putinar, S. Sullivant (eds.) *Emerging Applications of Algebraic Geometry*, Vol. 149 of *IMA Volumes in Mathematics and its Applications*, Springer-Verlag (2009)
65. Levenshtein, V.I.: Universal bounds for codes and designs. In: V. Pless, W. C. Huffman (eds.) *Handbook of Coding Theory*, Elsevier, Amsterdam (1998)

66. Linial, N., Magen, A., Naor, A.: Girth and Euclidean Distortion. *Geom. Funct. Anal.* **12**, 380–394 (2002)
67. Lovász, L.: On the Shannon capacity of a graph. *IEEE Trans. Inform. Theory* **25**, 1–5 (1979)
68. MacWilliams, F.J., Sloane, N.J.A.: The theory of error-correcting codes. North-Holland Mathematical Library, Vol. 16. North-Holland Publishing Co., Amsterdam-New York-Oxford (1977)
69. Martin, W.J., Stinson, D.R.: Association schemes for ordered orthogonal arrays and  $(T, M, S)$ -nets. *Canad. J. Math.* **51**, 326–346 (1999)
70. McEliece, R.J., Rodemich, E.R., Rumsey, Jr, H.C.: The Lovász bound and some generalizations. *J. Combinatorics, Inform. Syst. Sci.* **3**, 134–152 (1978)
71. Mittelmann, H.D., Vallentin, F.: High accuracy semidefinite programming bounds for kissing numbers. *Experiment. Math.* **19**, 174–178 (2010)
72. Murota, K., Kanno, Y., Kojima, M., Kojima, S.: A numerical algorithm for block-diagonal decomposition of matrix  $*$ -algebras, Part I: proposed approach and application to semidefinite programming. *Jpn. J. Ind. Appl. Math.* **27**, 125–160 (2010)
73. Musin, O.R.: Multivariate positive definite functions on spheres. arXiv:math/0701083 [math.MG] (2007)
74. Nemirovski, A.: Advances in convex optimization: Conic programming. In: M. Sanz-Sol, J. Soria, J.L. Varona, J. Verdera, (eds.) *Proceedings of International Congress of Mathematicians*, Madrid, August 22–30, 2006, Volume 1, European Mathematical Society Publishing House (2007)
75. Odlyzko, A.M., Sloane, N.J.A.: New bounds on the number of unit spheres that can touch a unit sphere in  $n$  dimensions. *J. Comb. Theory Ser. A* **26**, 210–214 (1979)
76. de Oliveira Filho, F.M.: New Bounds for Geometric Packing and Coloring via Harmonic Analysis and Optimization. Dissertation, University of Amsterdam (2009)
77. de Oliveira Filho, F.M., Vallentin, F.: Fourier analysis, linear programming, and densities of distance avoiding sets in  $\mathbb{R}^n$ . *J. Eur. Math. Soc. (JEMS)* **12**, 1417–1428 (2010)
78. Pasechnik, D.V., Kini, K.: A GAP package for computation with coherent configurations. In: K. Fukuda, J. van der Hoeven, M. Joswig, N. Takayama (eds.) *Mathematical Software — ICMS 2010 LNCS 6327*, Springer (2010)
79. Roy, A., Scott, A. J.: Unitary designs and codes. *Des. Codes Cryptogr.* **53**, 13–31 (2009)
80. Roy, A.: Bounds for codes and designs in complex subspaces. arXiv:0806.2317 [math.CO] (2008)
81. Rudin, W.: *Real and Complex Analysis*. McGraw-Hill International Editions (1987)
82. Schoenberg, I.J.: Positive definite functions on spheres. *Duke Math. J.* **9**, 96–108 (1942)
83. Schrijver, A.: Association schemes and the Shannon capacity: Eberlein polynomials and the Erdős-Ko-Rado theorem. *Algebraic methods in graph theory Vol. I, II* (Szeged, 1978), pp. 671–688, *Colloq. Math. Soc. János Bolyai* 25, North-Holland, Amsterdam-New York (1981)
84. Schrijver, A.: New code upper bounds from the Terwilliger algebra and semidefinite programming. *IEEE Trans. Inform. Theory* **51**, 2859–2866 (2005)
85. Soifer, A.: *The mathematical coloring book*. Springer-Verlag (2008)
86. Stanton, D.: Orthogonal polynomials and Chevalley groups. In: R.A. Askey, T.H. Koornwinder, W. Schempp (eds.) *Special functions: group theoretical aspects and applications*, Reidel Publishing Compagny (1984)
87. Sturm, J.F.: Using SeDuMi 1.02, A Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software* **11**, 625–653 (1999)
88. Sturmfels, B.: *Algorithms in Invariant Theory*. Springer-Verlag (1993)
89. Szegő, G.: *Orthogonal polynomials*. American Mathematical Society (1939)
90. Takesaki, M.: *Theory of Operator Algebras I*. Encyclopaedia of Mathematical Sciences, 124. *Operator Algebras and Non-commutative Geometry*, 5. Springer-Verlag, Berlin, (2002)
91. Tarnanen, H., Aaltonen, M., Goethals J.M.: On the nonbinary Johnson scheme. *European J. Combin.* **6**, 279–285 (1985)

92. Tarnanen, H.: Upper bounds on permutation codes via linear programming. *European J. Combin.* **20**, 101–114 (1999)
93. Vallentin, F.: Optimal embeddings of distance transitive graphs into Euclidean spaces. *J. Comb. Theory Ser. B* **98**, 95–104 (2008)
94. Vallentin, F.: Symmetry in semidefinite programs. *Linear Algebra and Appl.* **430**, 360–369 (2009)
95. Vallentin, F.: Lecture notes: Semidefinite programs and harmonic analysis. arXiv:0809.2017 [math.OC] (2008)
96. Vilenkin, N.Ja., Klimyk, A.U.: *Representation of Lie Groups and Special Functions, Volume 2*. Kluwer Academic Publishers (1993)
97. Wang, H.G.: Two-point homogeneous spaces. *Ann. of Math.* **55**, 177–191 (1952)
98. Woodall, D.R.: Cyclic-order graphs and Zarankiewicz’s crossing-number conjecture. *J. Graph Theory* **17**, 657–671 (1993)
99. Zarankiewicz, K.: On a problem of P. Turán concerning graphs. *Fundamenta Mathematicae* **41**, 137–145 (1954)

# Chapter 10

## A “Joint+Marginal” Approach in Optimization

Jean B. Lasserre

### 10.1 Introduction

The *moment approach* described in [15] is a powerful method to solve (or at least approximately solve) many important problems once they are viewed as a particular instance of the Generalized Problem of Moments (GPM in short). Among important applications, let us cite (global) optimization, probability and statistics, algebra, mathematical finance, control, games, inverse problems, applied mathematics and operations research, etc.

In particular, for polynomial (global) optimization, which is perhaps the simplest instance of the GPM, the moment approach consists of solving a *hierarchy* of convex relaxations (more precisely, semidefinite relaxations) of the initial problem, which provides a monotone nondecreasing sequence of lower bounds converging to the global optimum. In general, the convergence is fast and sometimes even finite, and in addition, when convergence is finite and a sufficient rank condition is satisfied, one may certify global optimality and extract global minimizers; for more details the interested reader is referred to e.g. [12, 15].

This chapter is about new developments of the moment approach for polynomial optimization, and more precisely, what we call the “joint+marginal” approach for *parametric* optimization. As a by-product, one also obtains a “joint+marginal” algorithm for *robust* polynomial optimization in a relatively general framework, as well as for the non parametric polynomial optimization problem.

---

J.B. Lasserre (✉)

LAAS-CNRS and Institute of Mathematics, University of Toulouse, 7 Avenue du Colonel Roche,  
31077 Toulouse Cédex 4, France

e-mail: [lasserre@laas.fr](mailto:lasserre@laas.fr)

### 10.1.1 Summary of the Contribution

Our contribution is twofold:

1. We first consider the parametric optimization problem:

$$\mathbf{P}_y : \quad J(y) = \min_{\mathbf{x}} \{f(\mathbf{x}, y) : \mathbf{x} \in \mathbf{K}_y\}$$

where  $\mathbf{y}$  is a *parameter* vector which lives in some parameter set  $\mathbf{Y} \subset \mathbb{R}^p$ ,  $f \in \mathbb{R}[\mathbf{x}, \mathbf{y}]$ , and for every  $\mathbf{y} \in \mathbf{Y}$ ,  $\mathbf{K}_y \subset \mathbb{R}^n$  is the basic semi-algebraic set

$$\mathbf{K}_y := \{\mathbf{x} \in \mathbb{R}^n : g_j(\mathbf{x}, \mathbf{y}) \geq 0, j = 1, \dots, m\},$$

for some polynomials  $(g_j) \subset \mathbb{R}[\mathbf{x}, \mathbf{y}]$ .

Computing the *optimal value function*  $\mathbf{y} \mapsto J(\mathbf{y})$  and/or the *global minimizer mapping*  $\mathbf{y} \mapsto \mathbf{x}^*(\mathbf{y})$  as functions of the parameter vector  $\mathbf{y}$ , is a very challenging infinite dimensional problem. Therefore, in general, a less ambitious and more realistic goal is to obtain information on  $J(\mathbf{y})$  and  $\mathbf{x}^*(\mathbf{y})$ , only locally around a nominal value  $\mathbf{y}_0$  of the parameter. However, in the polynomial context one shows that much more is possible. In particular, one can approximate the mappings  $J(\mathbf{y})$  and  $\mathbf{x}^*(\mathbf{y})$  via the “joint+marginal” approach which consists of solving a hierarchy of semidefinite relaxations, very much in the spirit of those developed in [12] for non parametric polynomial optimization.

2. We next use the preceding results to provide a “joint+marginal” algorithm:

- For the *robust* polynomial optimization problem:

$$J^* = \max_{\mathbf{y} \in \Omega} \min_{\mathbf{x}} \{f(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in \mathbf{K}_y\}, \quad (10.1)$$

which may be considered as a game against nature. Indeed, after  $\mathbf{y} \in \Omega$  (player1’s action) is chosen, player2 (nature) picks an action  $\mathbf{x}^*(\mathbf{y}) \in \mathbf{K}_y$  whose value  $J(\mathbf{y}) = f(\mathbf{x}^*(\mathbf{y}), \mathbf{y})$  minimizes  $f(\mathbf{x}, \mathbf{y})$  on  $\mathbf{K}_y$  (the worst possible outcome for  $\mathbf{y}$ ). As shown in Ben-tal et al. [2, 3], when the description of the uncertainty set takes some special form, one may obtain *computationally tractable* robust counterparts of some “easy” convex problems (like linear, second-order conic and semidefinite programs) which in general provide upper bounds on  $J^*$ , and even the exact value  $J^*$  in some special cases. Also, if  $f$  is affine in  $\mathbf{y}$  and  $\mathbf{K}_y = \mathbf{K}$  for all  $\mathbf{y} \in \Omega$ , then  $J^*$  can be approximated as closely as desired, by solving some appropriate hierarchy of semidefinite relaxations as described in [14]. But in the general context (10.1), robust optimization is a difficult and challenging problem for which there is still no general methodology.

However, notice that if on the one hand, (10.1) is a robust optimization problem for player1, on the other hand, it is a parametric optimization problem for player2. So a simple and natural idea is to apply to player2’s problem

the “joint+marginal” approach for parametric optimization so as to obtain a polynomial approximation  $p_k(\mathbf{y})$  of  $J(\mathbf{y})$ . Then player1 minimizes  $p_k(\mathbf{y})$  on  $\Omega$ , a non parametric polynomial optimization problem. We show that by increasing  $k$  one may approximate  $J^*$  as closely as desired, which to the best of our knowledge, seems to be the first result of this kind in this polynomial (but still quite general) context.

- For the non parametric optimization problem  $\mathbf{P} : \min_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{K}\}$ , when  $\mathbf{K} \subset \mathbb{R}^n$  is convex or when  $\mathbf{P}$  is a discrete optimization problem for which feasibility is easy to detect (like e.g., the MAXCUT,  $k$ -cluster and 0/1 knapsack problems). The basic idea is to consider  $x_1$  as a parameter in some interval  $\mathbf{Y}_1$  of the real line and approximate the univariate optimal value function  $J(y) = \min_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{K}; x_1 = y\}$  by a (univariate) polynomial  $J_k(y)$  (the higher the degree  $k$ , the better the approximation). Then one minimizes the univariate polynomial  $J_k$  over the interval  $\mathbf{Y}_1$  (by solving a single semidefinite program) to obtain some  $\tilde{x}_1 \in \mathbf{Y}$ . Next, in problem  $\mathbf{P}$ , fix  $x_1$  at the value  $\tilde{x}_1$  and iterate the procedure with now a new problem with the  $n - 1$  variables  $(x_2, x_3, \dots, x_n)$ . Again consider  $x_2$  a parameter in some interval  $\mathbf{Y}_2$  of the real line and compute  $\tilde{x}_2$ , etc., to finally obtain a vector  $\tilde{\mathbf{x}} = (\tilde{x}_1, \dots, \tilde{x}_n) \in \mathbf{K}$ . As a possible refinement, one may also use  $\tilde{\mathbf{x}}$  as the initial point of some standard minimization algorithm to obtain an even better local minimum  $\mathbf{x}^* \in \mathbf{K}$ .

## 10.2 Parametric Optimization

Let  $\mathbb{R}[\mathbf{x}, \mathbf{y}]$  denote the ring of real polynomials in the variables  $\mathbf{x} = (x_1, \dots, x_n)$  and  $\mathbf{y} = (y_1, \dots, y_p)$ . For a topological space  $\mathbf{X}$ , denote by  $\mathcal{B}(\mathbf{X})$  its associated Borel  $\sigma$ -field. Let  $\mathbf{Y} \subset \mathbb{R}^p$  be a compact set, called the *parameter* set, and let

$$\mathbf{K} := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{n+p} : \mathbf{y} \in \mathbf{Y}; \quad h_j(\mathbf{x}, \mathbf{y}) \geq 0, \quad j = 1, \dots, m\}, \quad (10.2)$$

for some polynomials  $(h_j) \subset \mathbb{R}[\mathbf{x}, \mathbf{y}]$ . For each  $\mathbf{y} \in \mathbf{Y}$ , fixed, consider the following optimization problem:

$$\mathbf{P}_{\mathbf{y}} : \quad J(\mathbf{y}) := \inf_{\mathbf{x}} \{f(\mathbf{x}, \mathbf{y}) : (\mathbf{x}, \mathbf{y}) \in \mathbf{K}\}. \quad (10.3)$$

The interpretation is as follows:  $\mathbf{Y}$  is a set of parameters and for each value  $\mathbf{y} \in \mathbf{Y}$  of the parameter, one wishes to compute an optimal *decision* vector  $\mathbf{x}^*(\mathbf{y})$  that solves problem (10.3). Parametric optimization is concerned with:

- The *optimal value function*  $\mathbf{y} \mapsto J(\mathbf{y})$ , as well as
- The set-valued *minimizer mapping*  $\mathbf{y} \mapsto \mathbf{X}^*(\mathbf{y})$  (resp. *dual multiplier mapping*  $\mathbf{y} \mapsto \Lambda^*(\mathbf{y})$ ) where for each  $\mathbf{y} \in \mathbf{Y}$ ,  $\mathbf{X}^*(\mathbf{y}) \subset \mathbb{R}^n$  (resp.  $\Lambda^*(\mathbf{y}) \subset \mathbb{R}^m$ ) is the set of global minimizers (resp. dual optimal solutions) of (10.3).

Of course, this is a difficult infinite-dimensional problem as one tries to approximate *functions*, and so in general, a less ambitious goal is to obtain information locally around some nominal value  $\mathbf{y}_0$  of the parameter. There is a vast and rich literature on the topic and for a detailed treatment, the interested reader is referred to e.g. Bonnans and Shapiro [6] and the many references therein. However, as one will see below, in the polynomial context, i.e., when  $f \in \mathbb{R}[\mathbf{x}, \mathbf{y}]$  and  $\mathbf{K}, \mathbf{Y}$  are basic semi-algebraic sets, much more is possible.

### 10.2.1 The “Joint+Marginal” Approach

Let  $\varphi$  be a Borel probability measure on  $\mathbf{Y}$ , with a positive density with respect to the Lebesgue measure on the smallest affine variety that contains  $\mathbf{Y}$ . For instance, choose for  $\varphi$  the probability measure uniformly distributed on  $\mathbf{Y}$ . Of course, one may also treat the case of a *discrete* set of parameters  $\mathbf{Y}$  (finite or countable) by taking for  $\varphi$  a discrete probability measure on  $\mathbf{Y}$  with strictly positive weight at each point of the support. Sometimes, e.g. in the context of optimization with data uncertainty,  $\varphi$  is already specified. We will use  $\varphi$  (or more precisely, its moments) to get information on the distribution of optimal solutions  $\mathbf{x}^*(\mathbf{y})$  of  $\mathbf{P}_{\mathbf{y}}$ , viewed as random vectors. In the rest of this section we assume that for every  $\mathbf{y} \in \mathbf{Y}$ , the set

$$\mathbf{K}_{\mathbf{y}} := \{\mathbf{x} \in \mathbb{R}^n : h_j(\mathbf{x}, \mathbf{y}) \geq 0, \quad j = 1, \dots, m\} \quad (10.4)$$

is not empty.

#### 10.2.1.1 A Related Infinite-Dimensional Linear Program

Let  $\mathbf{M}(\mathbf{K})$  be the set of finite Borel measures on  $\mathbf{K}$ , and consider the following infinite-dimensional linear program

$$\mathbf{P} : \quad \rho := \inf_{\mu \in \mathbf{M}(\mathbf{K})} \left\{ \int_{\mathbf{K}} f d\mu : \pi\mu = \varphi \right\} \quad (10.5)$$

where  $\pi\mu$  denotes the marginal of  $\mu$  on  $\mathbb{R}^p$ , that is,  $\pi\mu$  is a probability measure on  $\mathbb{R}^p$  defined by  $\pi\mu(B) = \mu(\mathbb{R}^n \times B)$  for all  $B \in \mathcal{B}(\mathbb{R}^p)$ . Notice that  $\mu(\mathbf{K}) = 1$  for any feasible solution  $\mu$  of  $\mathbf{P}$ . Indeed, as  $\varphi$  is a probability measure and  $\pi\mu = \varphi$  one has  $1 = \varphi(\mathbf{Y}) = \mu(\mathbb{R}^n \times \mathbb{R}^p) = \mu(\mathbf{K})$ .

**Theorem 10.1 ([16]).** *Let both  $\mathbf{Y} \subset \mathbb{R}^p$  and  $\mathbf{K}$  in (10.2) be compact and assume that for every  $\mathbf{y} \in \mathbf{Y}$ , the set  $\mathbf{K}_{\mathbf{y}} \subset \mathbb{R}^n$  in (10.4) is nonempty. For problem  $\mathbf{P}$  in (10.5), let  $\mathbf{X}^*(\mathbf{y}) := \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}, \mathbf{y}) = J(\mathbf{y})\}$ ,  $\mathbf{y} \in \mathbf{Y}$ . Then:*

- (a)  $\rho = \int_{\mathbf{Y}} J(\mathbf{y}) d\varphi(\mathbf{y})$  and  $\mathbf{P}$  has an optimal solution.

- (b) For every optimal solution  $\mu^*$  of  $\mathbf{P}$ , and for  $\varphi$ -almost all  $\mathbf{y} \in \mathbf{Y}$ , there is a probability measure  $\psi^*(d\mathbf{x}|\mathbf{y})$  on  $\mathbb{R}^n$ , concentrated on  $\mathbf{X}^*(\mathbf{y})$ , such that:

$$\mu^*(C \times B) = \int_B \psi^*(C|\mathbf{y}) d\varphi(\mathbf{y}), \quad \forall B \in \mathcal{B}(\mathbf{Y}), C \in \mathcal{B}(\mathbb{R}^n). \quad (10.6)$$

- (c) Assume that for  $\varphi$ -almost all  $\mathbf{y} \in \mathbf{Y}$ , the set of minimizers  $\mathbf{X}^*(\mathbf{y})$  is the singleton  $\{\mathbf{x}^*(\mathbf{y})\}$  for some  $\mathbf{x}^*(\mathbf{y}) \in \mathbf{K}_\mathbf{y}$ . Then there is a measurable mapping  $g : \mathbf{Y} \rightarrow \mathbf{K}_\mathbf{y}$  such that

$$g(\mathbf{y}) = \mathbf{x}^*(\mathbf{y}) \text{ for every } \mathbf{y} \in \mathbf{Y}; \quad \rho = \int_{\mathbf{Y}} f(g(\mathbf{y}), \mathbf{y}) d\varphi(\mathbf{y}), \quad (10.7)$$

and for every  $\alpha \in \mathbb{N}^n$ , and  $\beta \in \mathbb{N}^p$ :

$$\int_{\mathbf{K}} \mathbf{x}^\alpha \mathbf{y}^\beta d\mu^*(\mathbf{x}, \mathbf{y}) = \int_{\mathbf{Y}} \mathbf{y}^\beta g(\mathbf{y})^\alpha d\varphi(\mathbf{y}). \quad (10.8)$$

For a proof the interested reader is referred to [16]. As one may see, any optimal solution  $\mu^*$  of  $\mathbf{P}$  encodes *all* information on the global minimizers  $\mathbf{x}^*(\mathbf{y})$  of  $\mathbf{P}_\mathbf{y}$ . For instance, let  $\mathbf{B}$  be a given Borel set of  $\mathbb{R}^n$ . Then from Theorem 10.1,

$$\text{Prob}(\mathbf{x}^*(\mathbf{y}) \in \mathbf{B}) = \mu^*(\mathbf{B} \times \mathbb{R}^p) = \int_{\mathbf{Y}} \psi^*(\mathbf{B}|\mathbf{y}) d\varphi(\mathbf{y})$$

with  $\psi^*$  as in Theorem 10.1(b).

Now the reader can understand the name “joint+marginal”. Indeed, one considers a related linear programming problem  $\mathbf{P}$  over the space of Borel probability measures  $\mu$  (i.e. *joint* probability distributions on  $\mathbf{x}$  and  $\mathbf{y}$ ) whose *marginal* on  $\mathbf{Y}$  is fixed and equal to the given probability measure  $\varphi$ .

## Duality

Consider the following infinite-dimensional linear program  $\mathbf{P}^*$ :

$$\rho^* := \sup_{p \in \mathbb{R}[\mathbf{y}]} \left\{ \int_{\mathbf{Y}} p d\varphi : f(\mathbf{x}, \mathbf{y}) - p(\mathbf{y}) \geq 0 \quad \forall (\mathbf{x}, \mathbf{y}) \in \mathbf{K} \right\}. \quad (10.9)$$

Then  $\mathbf{P}^*$  is a *dual* of  $\mathbf{P}$  and the next result states that there is no duality gap between  $\mathbf{P}$  and  $\mathbf{P}^*$ .

**Lemma 10.1.** Let both  $\mathbf{Y} \subset \mathbb{R}^p$  and  $\mathbf{K}$  in (10.2) be compact and let  $\mathbf{P}$  and  $\mathbf{P}^*$  be as in (10.5) and (10.9), respectively. Then there is no duality gap, i.e.,  $\rho = \rho^*$ .

Moreover, any maximizing sequence of  $\mathbf{P}^*$  permits to approximate the optimal value function  $\mathbf{y} \mapsto J(\mathbf{y})$  in the following precise sense.

**Corollary 10.1.** Let both  $\mathbf{Y} \subset \mathbb{R}^p$  and  $\mathbf{K}$  in (10.2) be compact and assume that for every  $\mathbf{y} \in \mathbf{Y}$ , the set  $\mathbf{K}_\mathbf{y}$  in (10.4) is nonempty. Let  $\mathbf{P}^*$  be as in (10.9). If  $(p_i)_{i \in \mathbb{N}} \subset \mathbb{R}[\mathbf{y}]$  is a maximizing sequence of (10.9) then

$$\int_{\mathbf{Y}} |J(\mathbf{y}) - p_i(\mathbf{y})| d\varphi(\mathbf{y}) \rightarrow 0 \quad \text{as } i \rightarrow \infty. \quad (10.10)$$

Moreover, define the functions  $(\tilde{p}_i)$ ,  $i \in \mathbb{N}$ , as follows:

$$\tilde{p}_0 := p_0, \quad \mathbf{y} \mapsto \tilde{p}_i(\mathbf{y}) := \max [\tilde{p}_{i-1}(\mathbf{y}), p_i(\mathbf{y})], \quad i = 1, 2, \dots$$

Then  $\tilde{p}_i \rightarrow J(\cdot)$   $\varphi$ -almost uniformly<sup>1</sup> on  $\mathbf{Y}$ .

For a proof of Lemma 10.1 and Corollary 10.1, the reader is referred to [16].

Theorem 10.1 and Corollary 10.1 are even valid in the more general context where  $f$  is continuous and  $\mathbf{K}, \mathbf{Y}$  are compact sets. So, theoretically, any optimal solution of  $\mathbf{P}$  and its dual  $\mathbf{P}^*$  provide all information required in parametric optimization. However, in full generality,  $\mathbf{P}$  and  $\mathbf{P}^*$  are intractable numerically. But as one next will see, when the data are polynomials and basic semi-algebraic sets, one may obtain approximations of the optimal value function  $\mathbf{y} \mapsto J(\mathbf{y})$  as well as the global minimizer mapping  $\mathbf{y} \mapsto \mathbf{x}^*(\mathbf{y})$  (when the latter is well-defined).

## 10.2.2 A Hierarchy of Semidefinite Relaxations

### 10.2.2.1 Notation and Preliminaries

Let  $\Sigma[\mathbf{x}, \mathbf{y}] \subset \mathbb{R}[\mathbf{x}, \mathbf{y}]$  denote the set of polynomials that are sums of squares (s.o.s.), and let  $\mathbb{N}_i^n := \{\alpha \in \mathbb{N}^n : |\alpha| \leq i\}$  with  $|\alpha| = \sum_i \alpha_i$ . With a sequence  $\mathbf{z} = (z_{\alpha\beta})$ ,  $\alpha \in \mathbb{N}^n, \beta \in \mathbb{N}^p$ , indexed in the canonical basis  $(\mathbf{x}^\alpha \mathbf{y}^\beta)$  of  $\mathbb{R}[\mathbf{x}, \mathbf{y}]$ , let  $L_{\mathbf{z}} : \mathbb{R}[\mathbf{x}, \mathbf{y}] \rightarrow \mathbb{R}$  be the linear mapping:

$$f \left( = \sum_{\alpha\beta} f_{\alpha\beta} (\mathbf{x}, \mathbf{y}) \right) \mapsto L_{\mathbf{z}}(f) := \sum_{\alpha\beta} f_{\alpha\beta} z_{\alpha\beta}, \quad \forall f \in \mathbb{R}[\mathbf{x}, \mathbf{y}].$$

#### Moment Matrix

The moment matrix  $\mathbf{M}_i(\mathbf{z})$  associated with a sequence  $\mathbf{z} = (z_{\alpha\beta})$ , has its rows and columns indexed in  $\mathbb{N}_i^{n+p}$ , and with entries

---

<sup>1</sup>Recall that a sequence of measurable functions  $(h_n)$  on a measure space  $(\mathbf{Y}, \mathcal{B}(\mathbf{Y}), \varphi)$  converges to  $h$ ,  $\varphi$ -almost uniformly, if for every  $\epsilon > 0$  there is a set  $A \in \mathcal{B}(\mathbf{Y})$  such that  $\varphi(A) < \epsilon$  and  $h_n \rightarrow h$  uniformly on  $\mathbf{Y} \setminus A$ .

$$\mathbf{M}_i(\mathbf{z})(\alpha, \beta), (\delta, \gamma)) = L_{\mathbf{z}}(\mathbf{x}^\alpha \mathbf{y}^\beta \mathbf{x}^\delta \mathbf{y}^\gamma) = z_{(\alpha+\delta)(\beta+\gamma)},$$

for every  $(\alpha, \beta)$  and  $(\delta, \gamma)$  in  $\mathbb{N}_i^{n+p}$ .

### Localizing Matrix

Let  $q$  be the polynomial  $(\mathbf{x}, \mathbf{y}) \mapsto q(\mathbf{x}, \mathbf{y}) := \sum_{u,v} q_{uv} \mathbf{x}^u \mathbf{y}^v$ . The localizing matrix  $\mathbf{M}_i(q\mathbf{z})$  associated with  $q \in \mathbb{R}[\mathbf{x}, \mathbf{y}]$  and a sequence  $\mathbf{z} = (z_{\alpha\beta})$ , has its rows and columns indexed in  $\mathbb{N}_i^{n+p}$ , and with entries

$$\begin{aligned} \mathbf{M}_i(q\mathbf{z})(\alpha, \beta), (\delta, \gamma)) &= L_{\mathbf{z}}(q(\mathbf{x}, \mathbf{y}) \mathbf{x}^\alpha \mathbf{y}^\beta \mathbf{x}^\delta \mathbf{y}^\gamma) \\ &= \sum_{u \in \mathbb{N}^n, v \in \mathbb{N}^p} q_{uv} z_{(\alpha+\delta+u)(\beta+\gamma+v)}, \end{aligned}$$

for every  $(\alpha, \beta)$  and  $(\delta, \gamma)$  in  $\mathbb{N}_i^{n+p}$ .

A sequence  $\mathbf{z} = (z_{\alpha\beta}) \subset \mathbb{R}$  has a *representing* finite Borel measure supported on  $\mathbf{K}$  if there exists a finite Borel measure  $\mu$  such that

$$z_{\alpha\beta} = \int_{\mathbf{K}} \mathbf{x}^\alpha \mathbf{y}^\beta d\mu(\mathbf{x}, \mathbf{y}), \quad \forall \alpha \in \mathbb{N}^n, \beta \in \mathbb{N}^p.$$

The next important result states a necessary and sufficient condition for existence of a representing measure when  $\mathbf{K}$  is a compact basic semi-algebraic set whose defining polynomials  $(h_k) \subset \mathbb{R}[\mathbf{x}, \mathbf{y}]$  satisfy some condition.

**Assumption 10.1** Let  $(h_j)_{j=1}^t \subset \mathbb{R}[\mathbf{x}, \mathbf{y}]$  be a given family of polynomials. There is some  $N$  such that the quadratic polynomial  $(\mathbf{x}, \mathbf{y}) \mapsto N - \|(\mathbf{x}, \mathbf{y})\|^2$  can be written

$$N - \|(\mathbf{x}, \mathbf{y})\|^2 = \sigma_0 + \sum_{j=1}^t \sigma_j h_j,$$

for some s.o.s. polynomials  $(\sigma_j)_{j=0}^t \subset \Sigma[\mathbf{x}, \mathbf{y}]$ .

**Theorem 10.2 ([18]).** Let  $\mathbf{K} := \{(\mathbf{x}, \mathbf{y}) : h_k(\mathbf{x}, \mathbf{y}) \geq 0, k = 1, \dots, t\}$  and let  $(h_k)_{k=1}^t$  satisfy Assumption 10.1. A sequence  $\mathbf{z} = (z_{\alpha\beta})$  has a representing measure on  $\mathbf{K}$  if and only if, for all  $i = 0, 1, \dots$

$$\mathbf{M}_i(\mathbf{z}) \geq 0; \quad \mathbf{M}_i(h_k \mathbf{z}) \geq 0, \quad k = 1, \dots, t.$$

Theorem 10.2 is a direct consequence of Putinar’s Positivstellensatz [18] and [19]. Of course, when Assumption 10.1 holds then  $\mathbf{K}$  is compact. On the other hand, if

$\mathbf{K}$  is compact and one knows a bound  $N$  for  $\|(\mathbf{x}, \mathbf{y})\|$  on  $\mathbf{K}$ , then its suffices to add the redundant quadratic constraint  $h_{t+1}(\mathbf{x}, \mathbf{y})(:= N^2 - \|(\mathbf{x}, \mathbf{y})\|^2) \geq 0$  to the definition of  $\mathbf{K}$ , and Assumption 10.1 holds.

### 10.2.2.2 Semidefinite Relaxations

With  $\mathbf{K} \subset \mathbb{R}^n \times \mathbb{R}^p$  as in (10.2), let  $\mathbf{Y} \subset \mathbb{R}^p$  be the compact semi-algebraic set:

$$\mathbf{Y} := \{\mathbf{y} \in \mathbb{R}^p : h_k(\mathbf{y}) \geq 0, \quad k = m+1, \dots, t\} \quad (10.11)$$

for some polynomials  $(h_k)_{k=m+1}^t \in \mathbb{R}[\mathbf{y}]$ ; let  $v_k := \lceil (\deg h_k)/2 \rceil$  for every  $k = 1, \dots, t$ . Next, let  $\gamma = (\gamma_\beta)$  with

$$\gamma_\beta = \int_{\mathbf{Y}} \mathbf{y}^\beta d\varphi(\mathbf{y}), \quad \forall \beta \in \mathbb{N}^p, \quad (10.12)$$

be the moments of a probability measure  $\varphi$  on  $\mathbf{Y}$ , absolutely continuous with respect to the Lebesgue measure, and let  $i_0 := \max[\lceil (\deg f)/2 \rceil, \max_k v_k]$ . For  $i \geq i_0$ , consider the following semidefinite relaxations:

$$\begin{aligned} \rho_i &= \inf_{\mathbf{z}} L_{\mathbf{z}}(f) \\ \text{s.t. } &\mathbf{M}_i(\mathbf{z}) \succeq 0 \\ &\mathbf{M}_{i-v_j}(h_j \mathbf{z}) \succeq 0, \quad j = 1, \dots, t \\ &L_{\mathbf{z}}(\mathbf{y}^\beta) = \gamma_\beta, \quad \forall \beta \in \mathbb{N}_{2i}^p. \end{aligned} \quad (10.13)$$

**Theorem 10.3 ([16]).** Let  $\mathbf{K}, \mathbf{Y}$  be as in (10.2) and (10.11) respectively, and let  $(h_k)_{k=1}^t$  satisfy Assumption 10.1. Assume that for every  $\mathbf{y} \in \mathbf{Y}$  the set  $\mathbf{K}_\mathbf{y}$  in (10.4) is nonempty and consider the semidefinite relaxations (10.13). Then:

- (a)  $\rho_i \uparrow \rho$  as  $i \rightarrow \infty$ .
- (b) Let  $\mathbf{z}^i$  be a nearly optimal solution of (10.13), e.g. such that  $L_{\mathbf{z}^i}(f) \leq \rho_i + 1/i$ , and assume the same condition as in Theorem 10.1(c). Then with  $g : \mathbf{Y} \rightarrow \mathbf{K}_\mathbf{y}$  being the measurable mapping  $(g_1, \dots, g_n)$  in Theorem 10.1(c):

$$\lim_{i \rightarrow \infty} z_{\alpha\beta}^i = \int_{\mathbf{Y}} \mathbf{y}^\beta g(\mathbf{y})^\alpha d\varphi(\mathbf{y}), \quad \forall \alpha \in \mathbb{N}^n, \beta \in \mathbb{N}^p. \quad (10.14)$$

In particular, for every  $k = 1, \dots, n$ ,

$$\lim_{i \rightarrow \infty} z_{e(k)\beta}^i = \int_{\mathbf{Y}} \mathbf{y}^\beta \underbrace{g_k(\mathbf{y}) d\varphi(\mathbf{y})}_{d\phi_k(\mathbf{y})} = \int_{\mathbf{Y}} \mathbf{y}^\beta d\phi_k(\mathbf{y}), \quad \forall \beta \in \mathbb{N}^p, \quad (10.15)$$

where  $e(k)_j = \delta_{j=k}$ ,  $j = 1, \dots, n$  (with  $\delta$  being the Kronecker symbol).

Among other things, Theorem 10.3 states that if one let  $i \rightarrow \infty$  then one can approximate, as closely as desired, all moments of the (possibly signed) measures  $d\phi_k(\mathbf{y}) = g_k(\mathbf{y})d\mathbf{y}$  on  $\mathbf{Y}$ ,  $k = 1, \dots, n$  (recall that  $g(\mathbf{y}) = (g_1(\mathbf{y}), \dots, g_n(\mathbf{y}))$  is the global minimizer mapping of  $\mathbf{P}$ ). As one will see below, this precious moment information can be used to estimate the function  $g_k$  for every  $k = 1, \dots, n$ .

### 10.2.2.3 The Dual Semidefinite Relaxations

We now consider the dual of the semidefinite program (10.13) which reads:

$$\begin{aligned} \rho_i^* &= \sup_{p_i, (\sigma_j)} \int_{\mathbf{Y}} p_i d\varphi \quad \left( = \sum_{\beta \in \mathbb{N}_{2i}^B} p_{i\beta} \gamma_\beta \right) \\ \text{s.t. } f - p_i &= \sigma_0 + \sum'_{j=1} \sigma_j h_j \\ p_i &\in \mathbb{R}[\mathbf{y}]; \sigma_j \subset \Sigma[\mathbf{x}, \mathbf{y}], \quad j = 1, \dots, t \\ \deg p_i &\leq 2i, \deg \sigma_j h_j \leq 2i, \quad j = 1, \dots, t \end{aligned} \tag{10.16}$$

Observe that (10.16) is a strengthening of (10.9) as one restricts to polynomials  $p \in \mathbb{R}[\mathbf{y}]$  of degree at most  $2i$  and the nonnegativity of  $f - p$  in (10.9) is replaced with the stronger weighted s.o.s. representation in (10.16). Therefore  $\rho_i^* \leq \rho^*$  for every  $i$ .

**Theorem 10.4 ([16]).** *Let  $\mathbf{K}, \mathbf{Y}$  be as in (10.2) and (10.11) respectively, and let  $(h_k)_{k=1}^t$  satisfy Assumption 10.1. Assume that for every  $\mathbf{y} \in \mathbf{Y}$  the set  $\mathbf{K}_\mathbf{y}$  in (10.4) is nonempty, and consider the semidefinite relaxations (10.16). Then:*

- (a)  $\rho_i^* \uparrow \rho$  as  $i \rightarrow \infty$ .
- (b) Let  $(p_i, (\sigma_j))$  be a nearly optimal solution of (10.16), e.g. such that  $\int_{\mathbf{Y}} p_i d\varphi \geq \rho_i^* - 1/i$ . Then  $p_i(\cdot) \leq J(\cdot)$  and

$$\lim_{i \rightarrow \infty} \int_{\mathbf{Y}} |J(\mathbf{y}) - p_i(\mathbf{y})| d\varphi(\mathbf{y}) = 0. \tag{10.17}$$

Moreover, if one defines

$$\tilde{p}_0 := p_0, \quad \mathbf{y} \mapsto \tilde{p}_i(\mathbf{y}) := \max [\tilde{p}_{i-1}(\mathbf{y}), p_i(\mathbf{y})], \quad i = 1, 2, \dots,$$

then  $\tilde{p}_i \rightarrow J(\cdot)$   $\varphi$ -almost uniformly on  $\mathbf{Y}$ .

So if the primal provides useful information on the global minimizer mapping  $\mathbf{y} \mapsto g(\mathbf{y})$ , the dual semidefinite relaxations (10.16) provides lower polynomial (resp. piecewise polynomial) approximations  $(p_i(\mathbf{y}))_i$  (resp.  $(\tilde{p}_i)_i$ ) of the optimal value function  $J(\mathbf{y})$  of  $\mathbf{P}$ . Note that the  $L_1(\varphi)$ -norm convergence  $p_i \rightarrow J$  in (10.17) and the  $\varphi$ -almost uniform convergence  $\tilde{p}_i \rightarrow J$  both provide strong approximations of  $J$ .

### 10.2.3 Consequences

Theorem 10.3 has several interesting consequences.

#### 10.2.3.1 Functionals of Optimal Solutions

One may wish to evaluate the  $\varphi$ -mean or variance of optimal solutions  $\mathbf{x}^*(\mathbf{y})$  of  $\mathbf{P}_Y$ , or more generally some polynomial functional of  $\mathbf{x}^*(\mathbf{y})$ .

**Corollary 10.2 ([16]).** *Under the assumptions of Theorem 10.3, let  $h \in \mathbb{R}[\mathbf{x}]$  be the polynomial  $\mathbf{x} \mapsto h(\mathbf{x}) := \sum_{\alpha \in \mathbb{N}^n} h_\alpha \mathbf{x}^\alpha$ , and let  $\mathbf{z}^i$  be a nearly optimal solution of the semidefinite relaxations (10.13). Then:*

$$\lim_{i \rightarrow \infty} \sum_{\alpha \in \mathbb{N}^n} h_\alpha z_{\alpha 0}^i = \int_Y h(\mathbf{x}^*(\mathbf{y})) d\varphi(\mathbf{y}) = E_\varphi[h(\mathbf{x}^*(\mathbf{y}))].$$

#### 10.2.3.2 Persistence for Boolean Variables

Suppose that for some subset  $I \subseteq \{1, \dots, n\}$ , the variables  $(x_i)$ ,  $i \in I$ , are boolean, that is, the definition of  $\mathbf{K}$  in (10.2) includes the quadratic constraints  $x_i^2 - x_i = 0$ , for every  $i \in I$ . An interesting issue is to determine whether in any optimal solution  $\mathbf{x}^*(\mathbf{y})$  of  $\mathbf{P}_Y$ , and for some index  $i \in I$ , one has  $x_i^*(\mathbf{y}) = 1$  (or  $x_i^*(\mathbf{y}) = 0$ ) for  $\varphi$ -almost all values of the parameter  $\mathbf{y} \in Y$ . In [5, 17] the probability that  $x_i^*(\mathbf{y})$  is 1 is called the *persistency* of the boolean variable  $x_i^*(\mathbf{y})$ .

**Corollary 10.3 ([16]).** *Under the assumptions of Theorem 10.3, let  $\mathbf{z}^i$  be a nearly optimal solution of the semidefinite relaxations (10.13). Then for  $k \in I$  fixed:*

- (a)  $x_k^*(\mathbf{y}) = 1$  in any optimal solution and for  $\varphi$ -almost all  $\mathbf{y} \in Y$ , only if  $\lim_{i \rightarrow \infty} z_{e(k)0}^i = 1$ .
  - (b)  $x_k^*(\mathbf{y}) = 0$  in any optimal solution and for  $\varphi$ -almost all  $\mathbf{y} \in Y$ , only if  $\lim_{i \rightarrow \infty} z_{e(k)0}^i = 0$ .
- Assume that for  $\varphi$ -almost all  $\mathbf{y} \in Y$ ,  $J(\mathbf{y})$  is attained at a unique optimal solution  $\mathbf{x}^*(\mathbf{y}) \in \mathbf{X}_Y^*$ . Then  $\text{Prob}(x_k^*(\mathbf{y}) = 1) = \lim_{i \rightarrow \infty} z_{e(k)0}^i$ , and so:
- (c)  $x_k^*(\mathbf{y}) = 1$  for  $\varphi$ -almost all  $\mathbf{y} \in Y$ , if and only if  $\lim_{i \rightarrow \infty} z_{e(k)0}^i = 1$ .
  - (d)  $x_k^*(\mathbf{y}) = 0$  for  $\varphi$ -almost all  $\mathbf{y} \in Y$ , if and only if  $\lim_{i \rightarrow \infty} z_{e(k)0}^i = 0$ .

To explain the difference between (a) and (c) (as well as between (b) and (d)), suppose that for every  $\mathbf{y} \in Y$  there is an optimal solution  $\mathbf{x}^*(\mathbf{y}) \in \mathbf{X}^*(\mathbf{y})$  with  $x_k^*(\mathbf{y}) = 1$ , and there are also other optimal solutions  $\hat{\mathbf{x}}(\mathbf{y}) \in \mathbf{X}^*(\mathbf{y})$ , where  $\hat{x}_k(\mathbf{y}) = 0$  whenever  $\mathbf{y} \in A$  (where  $A \subset Y$  and  $\varphi(A) > 0$ ). So, there is an optimal probability measure  $\mu^*$  for problem (10.5) where in (10.6),  $\psi^*(d\mathbf{x}|\mathbf{y}) = \delta_{\mathbf{x}^*(\mathbf{y})}$  with  $\mathbf{x}^*(\mathbf{y}) \in \mathbf{X}^*(\mathbf{y})$  and  $x_k^*(\mathbf{y}) = 1$

for all  $\mathbf{y} \in \mathbf{Y}$ . If  $\mathbf{z}^i$  converges to the sequence of moments of  $\mu^*$  then  $\lim_{i \rightarrow \infty} z_{e(k)0}^i = 1$  whereas one may have  $\hat{x}_k(\mathbf{y}) = 0$ ,  $\mathbf{y} \in A$ , in some optimal solutions  $\hat{\mathbf{x}}(\mathbf{y})$ .

### 10.2.4 Estimating the Global Minimizer Mapping

By Corollary 10.2, one may approximate any polynomial functional of the optimal solutions. But one may also wish to approximate (in some sense) the function  $\mathbf{y} \mapsto \mathbf{x}_k^*(\mathbf{y})$ , that is, the “curve” described by the  $k$ -th coordinate  $\mathbf{x}_k^*(\mathbf{y})$  of the optimal solution  $\mathbf{x}^*(\mathbf{y})$  when  $\mathbf{y}$  varies in  $\mathbf{Y}$ .

So let  $g : \mathbf{Y} \rightarrow \mathbb{R}^n$  be the measurable mapping in Theorem 10.1(c) and suppose that one knows some lower bound vector  $\mathbf{a} = (a_k) \in \mathbb{R}^n$ , where:

$$a_k \leq \inf \{x_k : (\mathbf{x}, \mathbf{y}) \in \mathbf{K}\}, \quad k = 1, \dots, n.$$

Then for every  $k = 1, \dots, n$ , the measurable function  $\hat{g}_k : \mathbf{Y} \rightarrow \mathbb{R}^n$  defined by

$$\mathbf{y} \mapsto \hat{g}_k(\mathbf{y}) := g_k(\mathbf{y}) - a_k, \quad \mathbf{y} \in \mathbf{Y}, \quad (10.18)$$

is nonnegative and  $\varphi$ -integrable. Hence for every  $k = 1, \dots, n$ , one may consider  $d\lambda := \hat{g}_k d\varphi$  as a Borel measure on  $\mathbf{Y}$  with unknown density  $\hat{g}_k$  with respect to  $\varphi$ , but with known moments  $\mathbf{u} = (u_\beta)$ . Indeed, using (10.15),

$$\begin{aligned} u_\beta &:= \int_{\mathbf{Y}} \mathbf{y}^\beta d\lambda(\mathbf{y}) = -a_k \int_{\mathbf{Y}} \mathbf{y}^\beta d\varphi(\mathbf{y}) + \int_{\mathbf{Y}} \mathbf{y}^\beta g_k(\mathbf{y}) d\varphi(\mathbf{y}) \\ &= -a_k \gamma_\beta + z_{e(k)\beta}, \quad \forall \beta \in \mathbb{N}^p, \end{aligned} \quad (10.19)$$

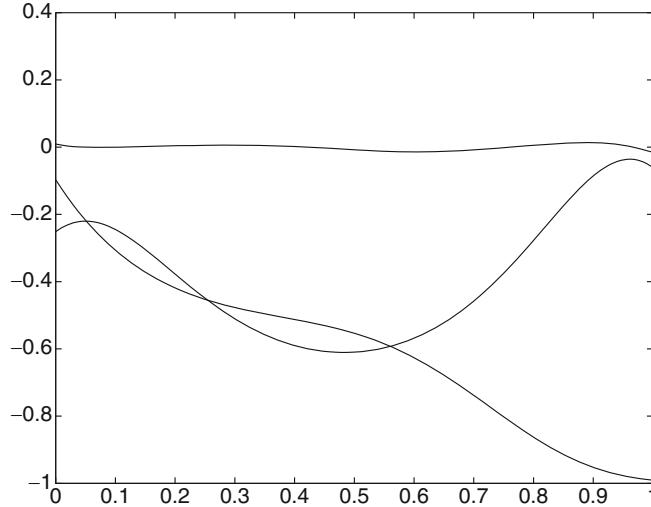
where for every  $k = 1, \dots, n$ ,  $z_{e(k)\beta} = \lim_{i \rightarrow \infty} z_{e(k)\beta}^i$  for all  $\beta \in \mathbb{N}^p$ , with  $\mathbf{z}^i$  being an optimal (or nearly optimal) solution of the semidefinite relaxation (10.13).

Hence we are now faced with a density estimation problem, that is: Given the sequence of moments  $u_\beta = \int_{\mathbf{Y}} \mathbf{y}^\beta \hat{g}_k(\mathbf{y}) d\varphi(\mathbf{y})$ ,  $\beta \in \mathbb{N}^p$ , of the unknown nonnegative measurable function  $\hat{g}_k$  on  $\mathbf{Y}$ , “estimate”  $\hat{g}_k$ . One possibility is to use the *maximum entropy* approach where given finitely many moments  $\mathbf{u} = (u_\beta)$  of  $\hat{g}_k$  on  $\mathbf{Y}$ , one obtains  $\hat{g}_k$  as the optimal solution of a convex optimization problem as described in e.g. [16].

For instance, suppose that  $\mathbf{Y} = [0, 1]$  and  $2d+1$  moments  $\mathbf{u} = (\mathbf{u}_j)_{j=0}^{2d}$  are available. Using the Boltzmann–Shanon entropy, one searches for an optimal estimate of the form  $h_d^*(y) = \exp \sum_{j=0}^{2d} \lambda_j^* y^j$ , where  $\lambda^* \in \mathbb{R}^{2d+1}$  is an optimal solution of the convex optimization problem

$$\sup_{\lambda \in \mathbb{R}^{2d+1}} \left\{ \lambda' \mathbf{u} - \int_0^1 \exp \left( \sum_{k=0}^{2d} \lambda_k x^k \right) dx \right\}. \quad (10.20)$$

For more detail the interested reader is referred to e.g. [16] and the references therein.



**Fig. 10.1** Example 10.1:  $x_1^*(y)$ ,  $x_2^*(y)$  and  $h_1(\mathbf{x}^*(y), y)$  on  $[0, 1]$

*Example 10.1.* In this example,  $\mathbf{Y} = [0, 1]$ ,  $(\mathbf{x}, y) \mapsto f(\mathbf{x}, y) := yx_1 + (1-y)x_2$ , and  $\mathbf{K} := \{(\mathbf{x}, y) : yx_1^2 + x_2^2 - y \leq 0; x_1^2 + yx_2^2 - y \leq 0\}$ . That is, for each  $y \in \mathbf{Y}$  the set  $\mathbf{K}_y$  is the intersection of two ellipsoids. It is easy to check that  $1 + x_i^*(y) \geq 0$  for all  $y \in \mathbf{Y}$ ,  $i := 1, 2$ .

With  $d=2$  (i.e., 4 moments), the max-entropy estimate  $h_4^*(y)$  for  $1 + x_1^*(y)$  reads  $y \mapsto h_4^*(y) = \exp \sum_{j=0}^4 \lambda_j^* y^j$ , and is obtained with the optimal solution  $\lambda^* = (-0.2894, 1.7192, -19.8381, 36.8285, -18.4828)$  of (10.20). Similarly, the max-entropy estimate  $h_4^*(y)$  for  $1 + x_2^*(y)$  is obtained with  $\lambda^* = (-0.1018, -3.0928, 4.4068, 1.7096, -7.5782)$ .

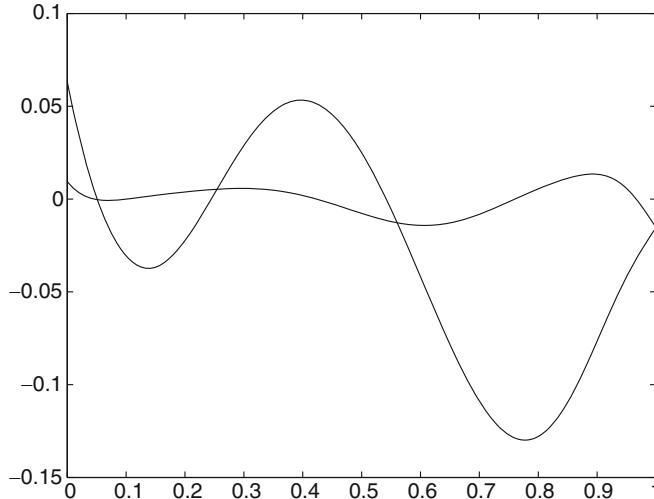
Figure 10.1 displays the curves of  $x_1^*(y)$  and  $x_2^*(y)$ , as well as the constraint  $h_1(\mathbf{x}^*(y), y)$ . Observe that  $h_1(\mathbf{x}^*(y), y) \approx 0$  on  $[0, 1]$  which means that for  $\varphi$ -almost all  $y \in [0, 1]$ , at an optimal solution  $\mathbf{x}^*(y)$ , the constraint  $h_1 \leq 0$  is saturated. Figure 10.2 displays the curves of  $h_1(\mathbf{x}^*(y), y)$  and  $h_2(\mathbf{x}^*(y), y)$ .

### 10.3 Robust Polynomial Optimization

Let  $\mathbf{K} \subset \mathbb{R}^{n+p}$  be the basic semi-algebraic set defined in (10.2) and let  $\mathbf{K}_y$  be as in (10.4). In this section we are interested in the min-max optimization problem:

$$\mathbf{PR} : \quad J^* = \max_{\mathbf{y} \in \Omega} \min_{\mathbf{x}} \{f(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in \mathbf{K}_y\}, \quad (10.21)$$

where  $\Omega \subset \mathbb{R}^p$  is a compact basic semi-algebraic set.



**Fig. 10.2** Example 10.1:  $h_1(\mathbf{x}^*(y), y)$  and  $h_2(\mathbf{x}^*(y), y)$  on  $[0, 1]$

Problem **PR** can be viewed as a *game* problem where  $\Omega$  is player1's action set and  $\mathbf{K}_y$  is player2's action set, once player1 has chosen action  $y \in \Omega$ . So from the point of view of player1, **PR** is a *robust optimization* problem since an optimal action  $y^* \in \Omega$  has to be made *before* player2 chooses an action  $\mathbf{x}^*(y^*)$  that minimizes his objective function  $\mathbf{x} \mapsto f(\mathbf{x}, y^*)$ . Hence player1 has to choose an action which is the best possible against the worst action by player2. If one tolerates mixed strategies (i.e. each player may choose a probability distribution on his set of available actions) then under some conditions, there exists a Nash equilibrium and one may interchange the min and max operators.

As already mentioned, in the general context (10.21), and without further assumptions on the data  $\Omega, \mathbf{K}_y, f$ , and the description of the uncertainty set, **PR** is a difficult and challenging problem for which there is no general methodology. On the other hand, from the point of view of player2, **PR** is parametric optimization problem because player2 waits for the decision  $y$  of player1 and then chooses the best possible action  $\mathbf{x}^*(y)$  that minimizes  $f(\mathbf{x}, y)$ . Thus, observe that if player1 knows the optimal value function  $y \mapsto J(y) = \min_{\mathbf{x}} \{f(\mathbf{x}, y) : \mathbf{x} \in \mathbf{K}_y\}$ , then **PR** reduces to the optimization problem  $J^* = \max_y \{J(y) : y \in \Omega\}$ . If in general there is no closed-form expression for  $J$ , we have just seen in Sect. 10.2 that in some polynomial context one may approximate (in a strong sense) the optimal value function  $J(\cdot)$  by polynomials or piecewise polynomials; see Theorem 10.4.

So we propose to replace **PR** with the optimization problem  $\max_y \{p_i(y) : y \in \Omega\}$ , where  $p_i$  is some polynomial approximation of the optimal value function  $y \mapsto J(y)$ . Inspired by results of Sect. 10.2, we need a set  $\mathbf{Y} \subset \mathbb{R}^p$  that contains  $\Omega$  and simple enough so that one may compute easily all moments of a probability measure  $\varphi$  on  $\mathbf{Y}$ , absolutely continuous with respect to the Lebesgue measure on the smallest affine

variety that contains  $\Omega$ . In general  $\Omega$  is a complicated set and so the choice  $\mathbf{Y} = \Omega$  is not appropriate (except if  $\Omega$  is a basic semi-algebraic set of small dimension, in which case the moments of  $\varphi$  can be approximated as described in [9]).

So we assume that  $\Omega \subseteq \mathbf{Y}$  where  $\mathbf{Y}$  is a box or an ellipsoid, and  $\varphi$  is the probability with uniform distribution on  $\mathbf{Y}$ , so that all its moments ( $\gamma_\beta$ ) in (10.12) can be computed.

**Proposition 10.1.** *Under the assumptions of Theorem 10.3, the optimal value function  $\mathbf{y} \mapsto J(\mathbf{y})$  is lower semicontinuous (l.s.c.) on  $\mathbf{Y}$ .*

*Proof.* Let  $(n_\ell)$  be a sequence such that  $\mathbf{y}_{n_\ell} \in \mathbf{Y}$  for every  $\ell$  and

$$\liminf_{z \rightarrow y} J(z) = \lim_{\ell \rightarrow \infty} J(\mathbf{y}_{n_\ell}).$$

Next let  $\mathbf{x}^*(\mathbf{y}_{n_\ell}) \in \mathbf{X}^*(\mathbf{y}_{n_\ell})$  be an arbitrary minimizer in  $\mathbf{K}_{\mathbf{y}_{n_\ell}}$ . By compactness of  $\mathbf{K}$ , there is a subsequence (still denoted  $(n_\ell)$  for notational simplicity) and a point  $(\mathbf{a}, \mathbf{y}) \in \mathbf{K}$  such that  $(\mathbf{x}^*(\mathbf{y}_{n_\ell}), \mathbf{y}_{n_\ell}) \rightarrow (\mathbf{a}, \mathbf{y})$ . Consequently,

$$\begin{aligned} \liminf_{z \rightarrow y} J(z) &= \lim_{\ell \rightarrow \infty} J(\mathbf{y}_{n_\ell}) = \lim_{\ell \rightarrow \infty} f(\mathbf{x}^*(\mathbf{y}_{n_\ell}), \mathbf{y}_{n_\ell}) \\ &= f(\mathbf{a}, \mathbf{y}) \geq \min_{\mathbf{x}} \{f(\mathbf{x}, \mathbf{y}) : (\mathbf{x}, \mathbf{y}) \in \mathbf{K}\} = J(\mathbf{y}), \end{aligned}$$

which proves that  $J$  is l.s.c.  $\square$

**Theorem 10.5.** *Let  $\Omega \subseteq \mathbf{Y}$  and under the assumptions of Theorem 10.3, let  $p_i \in \mathbb{R}[\mathbf{y}]$  be the polynomial defined in Theorem 10.4(b). Consider the optimization problem:*

$$\mathbf{PR}_i : \quad \max_{\mathbf{y}} \{p_i(\mathbf{y}) : \mathbf{y} \in \Omega\}, \quad (10.22)$$

with optimal value  $J_i^*$ . Let  $\mathbf{y}_i^* \in \Omega$  be some global minimizer of  $\mathbf{PR}_i$  and for every  $i$ , let  $\hat{J}_i^* := \sup_{\ell=1,\dots,i} J_\ell^* = J_{t(i)}(\mathbf{y}_{t(i)}^*)$  for some  $t(i) \in \{1, \dots, i\}$ .

- (a) Then  $\hat{J}_i^* \rightarrow J^*$  as  $i \rightarrow \infty$ .
- (b) If  $J$  is continuous, so that  $J^* = J(\mathbf{y}^*)$  for some  $\mathbf{y}^* \in \Omega$ , then any accumulation point  $\bar{\mathbf{y}}$  of the sequence  $(\mathbf{y}_{t(i)}^*) \subset \Omega$  is a global minimizer of  $\mathbf{PR}$ . In particular, if  $\mathbf{y}^*$  is the unique maximizer of  $\mathbf{PR}$  then  $\mathbf{y}_{t(i)}^* \rightarrow \mathbf{y}^*$  as  $i \rightarrow \infty$ .

*Proof.* (a) Observe that  $p_i$  being continuous on  $\mathbf{Y}$  (hence on  $\Omega$ ),  $\mathbf{PR}_i$  has a global minimizer  $\mathbf{y}_i^*$ , for every  $i$ . Next, recall that from Theorem 10.4 (b),  $p_i \xrightarrow{L_1(\varphi)} J$  (i.e., convergence in the  $L_1(\varphi)$ -norm). Hence, by [1, Theorem 2.5.1],  $p_i \xrightarrow{\varphi} J$  (i.e. convergence in probability) and in turn, by [1, Theorem 2.5.3], there exists a subsequence  $(i_\ell)$  such that  $p_{i_\ell} \rightarrow J$ ,  $\varphi$ -almost uniformly.

Next, by Proposition 10.1, the optimal value mapping  $J$  is l.s.c. on  $\mathbf{Y}$  (hence on  $\Omega \subseteq \mathbf{Y}$ ). With  $\epsilon > 0$  fixed, arbitrary, let  $\mathbf{B}(\epsilon) := \{\mathbf{y} \in \Omega : J(\mathbf{y}) > J^* - \epsilon\}$  and

let  $\kappa := \varphi(\mathbf{B}(\epsilon))$ . As  $J$  is l.s.c.,  $B(\epsilon)$  is nonempty, open, and therefore  $\kappa > 0$ . As  $p_{i_\ell} \rightarrow J$ ,  $\varphi$ -almost uniformly on  $\mathbf{Y}$ , there exists a Borel set  $A_\kappa \in \mathcal{B}(\mathbf{Y})$  such that  $\varphi(A_\kappa) < \kappa$  and  $p_{i_\ell} \rightarrow J$ , uniformly on  $\mathbf{Y} \setminus A_\kappa$ . Hence, as  $\mathcal{A} := (\mathbf{Y} \setminus A_\kappa) \cap \mathbf{B}(\epsilon) \neq \emptyset$ , one has

$$\lim_{\ell \rightarrow \infty} p_{i_\ell}(\mathbf{y}) = J(\mathbf{y}) \geq J^* - \epsilon, \quad \forall \mathbf{y} \in \mathcal{A},$$

and so, as  $J_i^* \geq p_i(\mathbf{y})$  on  $\mathcal{A}$ , one obtains  $\lim_{\ell \rightarrow \infty} J_{i_\ell}^* \geq J^* - \epsilon$ . As  $\epsilon > 0$  was arbitrary, one finally gets  $\lim_{\ell \rightarrow \infty} J_{i_\ell}^* = J^*$ . On the other hand, by monotonicity of the sequence  $(\hat{J}_i^*)$ ,

$$J^* \geq \lim_{i \rightarrow \infty} \hat{J}_i^* = \lim_{\ell \rightarrow \infty} \hat{J}_{i_\ell}^* \geq \lim_{\ell \rightarrow \infty} J_{i_\ell}^* = J^*,$$

and so the conclusion of (a) follows.

- (b) Let  $\mathbf{y}_i^* \in \Omega$  be a maximizer of  $\mathbf{PR}_i$ . As  $\Omega$  is compact, there exists  $\bar{\mathbf{y}} \in \Omega$  and a subsequence  $i_\ell$  such that  $\mathbf{y}_{t(i_\ell)}^* \rightarrow \bar{\mathbf{y}}$  as  $\ell \rightarrow \infty$ . In addition, from  $p_i(\mathbf{y}) \leq J(\mathbf{y})$  for every  $i$  and every  $\mathbf{y} \in \mathbf{Y}$ ,

$$J^* \geq J(\mathbf{y}_{t(i_\ell)}^*) \geq p_{t(i_\ell)}(\mathbf{y}_{t(i_\ell)}^*) = \hat{J}_{i_\ell}^*.$$

So using (a) and letting  $\ell \rightarrow \infty$  yields the desired result  $J^* = J(\bar{\mathbf{y}})$ . Since the converging subsequence  $(i_\ell)$  was arbitrary, the desired result follows.  $\square$

Continuity of  $J$  is restrictive and is obtained if, e.g., the set-valued mapping  $\mathbf{y} \mapsto \mathbf{X}^*(\mathbf{y})$  l.s.c., that is,  $(\mathbf{X}^*)^{-1}(O)$  is open whenever  $O \subset \mathbb{R}^n$  is open; see e.g. Proposition D 6(b) in [11, p. 183].

Hence Theorem 10.5 states that one may approximate as closely as desired the optimal value  $J^*$  of the robust optimization problem  $\mathbf{PR}$  by minimizing the polynomial  $p_i$  provided that  $i$  is sufficiently large. Of course, for robust optimization problems with specific features, like some described in [2, 3], we do not claim that this algorithm might compete with other algorithms tailored to such cases. However, and to the best of our knowledge, in the polynomial (but still quite large) context, it seems to be the first with guaranteed convergence.

*Example 10.2.* In this example taken from [4, Sect. 4.1],  $\Omega = \mathbf{Y} = [-1, 1]^2 \subset \mathbb{R}^2$  and  $\mathbf{K}_y = \{\mathbf{x} \in \mathbb{R}^2 : \|\mathbf{x}\| \leq 0.5\}$ . The robust problem (10.21) reads:

$$\mathbf{PR} : \min_{\mathbf{y} \in \Omega} \max_{\mathbf{x}} \{f(\mathbf{x} + \mathbf{y}) : \|\mathbf{x}\|^2 \leq 0.25\},$$

where  $f \in \mathbb{R}[\mathbf{x}]$  is the degree 6 polynomial

$$\begin{aligned} \mathbf{x} \mapsto f(\mathbf{x}) = & 2x_1^6 - 12.2x_1^5 + 21.2x_1^4 - 6.4x_1^3 - 4.7x_1^2 + 6.2x_1 + x_2^6 - 11x_2^5 + 43.3x_2^4 \\ & - 74.8x_2^3 + 56.9x_2^2 - 10x_2 - 0.1x_1^2x_2^2 + 0.4x_1^2x_2 + 0.4x_1x_2^2 - 4.1x_1x_2. \end{aligned}$$

**Table 10.1** Comparing  $p_i$  with  $J$  on a 400-point grid  $G \subset [-1, 1]^2$ 

	$e(p_i, J)$	$e(\tilde{p}_i, J)$	$J_i^*$	$\mathbf{y}_i^*$	$J(\mathbf{y}_i^*)$
$i = 3$	7.07%	7.07%	12.59	(0.0300, 0.3617)	11.91
$i = 4$	4.70%	4.46%	12.12	(-0.0036, 0.3347)	11.49
$i = 5$	3.53%	3.27%	11.74	(-0.0553, 0.3112)	11.07

With  $i = 3, 4, 5$ , we have computed the polynomial  $p_i \in \mathbb{R}[\mathbf{y}]$  in Theorem 10.5 as well as the piecewise polynomial  $\mathbf{y} \mapsto \tilde{p}_i(\mathbf{y}) := \max\{p_\ell(\mathbf{y}) : \ell = 3, \dots, i\}$ . We have compared the values of  $p_i(\mathbf{y})$  and  $\tilde{p}_i(\mathbf{y})$  with

$$J(\mathbf{y}) := \max\{f(\mathbf{x}, \mathbf{y}) : \|\mathbf{x}\|^2 \leq 0.25\},$$

on a 400-point grid  $G \subset [-1, 1]^2$ . Denote by  $e(p_i, J)$  (resp.  $e(\tilde{p}_i, J)$ ) the average relative error on the grid  $G$  when comparing  $p_i$  (resp.  $\tilde{p}_i$ ) with  $J$ . Finally, let  $\mathbf{y}_i^* \in \mathbf{Y}$  denote the global optimum of  $p_i$  on  $\mathbf{Y}$ .

As one may see in Table 10.1, and despite  $f$  is a degree 6 polynomial, one already obtains a quite small relative error with moments up to order 8 only.

## 10.4 A “Joint+Marginal” Algorithm in Optimization

Consider the (non parametric) polynomial program

$$\mathbf{P} : \quad f^* := \min_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{K}\} \tag{10.23}$$

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

$$\mathbf{K} := \{\mathbf{x} \in \mathbb{R}^n : h_j(\mathbf{x}) \geq 0, \quad j = 1, \dots, m\}, \tag{10.24}$$

for some polynomials  $(h_j) \subset \mathbb{R}[\mathbf{x}]$ , and  $f^*$  is the *global* minimum of  $\mathbf{P}$  (as opposed to a local minimum). One way to approximate  $f^*$  is to solve a hierarchy of either LP-relaxations or semidefinite relaxations as proposed in e.g. Lasserre [12, 13]. Despite practice with the semidefinite relaxations seems to reveal that convergence is fast, the matrix size in the  $i$ -th semidefinite relaxation of the hierarchy grows up as fast as  $O(n^i)$ . Hence, for large size (and sometimes even medium size) problems, only a few relaxations of the hierarchy can be implemented (typically the first, second or third relaxation). In that case, one only obtains a lower bound on the optimal value  $f^*$ , and no feasible solution.

So an important issue is: *How can we use the result of the  $i$ -th semidefinite relaxation to find an approximate solution of the original problem?*

For some well-known special cases in 0/1 optimization like e.g. the celebrated MAXCUT problem, one may generate a feasible solution with guaranteed

performance, from a randomized procedure that uses an optimal solution of the first semidefinite relaxation (i.e. with  $i = 1$ ); see Goemans and Williamson [8]. But in general there is no such procedure.

### 10.4.1 The “Joint+Marginal” Strategy

In this section we provide a relatively simple algorithm for polynomial programs with a convex feasible set  $\mathbf{K}$  and for 0/1 programs for which feasibility is easy to detect, like e.g., the MAXCUT,  $k$ -CLUSTER or 0/1 knapsack problems. The algorithm builds up upon the “joint+marginal” approach (in short (J+M)) developed in Sect. 10.2 for *parametric* polynomial optimization.

Let the compact interval  $\mathbf{Y}_1 := [\underline{x}_1, \bar{x}_1] \subset \mathbb{R}$  be the projection of  $\mathbf{K}$  on the  $x_1$ -coordinate axis, and consider the parametric optimization problem:

$$\mathbf{P}(x_1) : \quad J^1(y) = \min_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{K}; x_1 = y\}, \quad (10.25)$$

or, equivalently  $J^1(y) = \min_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{K}_y\}$ . Observe that (10.25) is a particular case of the parametric problem  $\mathbf{P}_y$  in (10.3) with  $\mathbf{Y} = [\underline{x}_1, \bar{x}_1]$ , and  $\mathbf{K}_y \neq \emptyset$  whenever  $y \in \mathbf{Y}_1$ . Next, by definition,  $f^* = \min_x \{J^1(x) : x \in \mathbf{Y}_1\}$ .

In the context of the (non-parametric) polynomial optimization (10.23), the (J+M)-approach of Sect. 10.2 can be used as follows in what we call the **(J+M)-algorithm**: Let  $i \in \mathbb{N}$  be fixed.

- (a) Treat  $x_1$  as a parameter in the compact interval  $\mathbf{Y}_1 = [\underline{x}_1, \bar{x}_1]$  with associated probability distribution  $\varphi_1$ , uniformly distributed on  $\mathbf{Y}_1$ . (For 0/1 programs  $\mathbf{Y}_1 = \{0, 1\}$ .)
- (b) Recall that  $i \in \mathbb{N}$  is fixed. Solve the  $i$ -th semidefinite relaxation of the (J+M)-hierarchy (10.13) applied to the parametric optimization problem (10.25) with  $n - 1$  variables  $(x_2, \dots, x_n)$  and parameter  $x_1$ . The dual (10.16) provides a univariate polynomial  $x_1 \mapsto p_i^1(x_1)$  that converges to  $J^1$  in the  $L_1(\varphi_1)$ -norm, as  $i$  increases. (The map  $v \mapsto J^1(v)$  denotes the optimal value function of  $\mathbf{P}(v)$ , i.e. the optimal value of  $\mathbf{P}$  given that the variable  $x_1$  is fixed at the value  $v$ .) Compute a global minimizer  $\tilde{x}_1 \in \mathbf{Y}_1$  of the univariate polynomial  $p_i^1$  on  $\mathbf{Y}_1$  (e.g. this can be done by solving a single semidefinite program). Ideally, when  $i$  is large enough,  $\tilde{x}_1$  should be close to the first coordinate  $x_1^*$  of a global minimizer  $\mathbf{x}^* = (x_1^*, \dots, x_n^*)$  of  $\mathbf{P}$ .
- (c) consider now the new optimization problem  $\mathbf{P}(\tilde{x}_1)$  in the  $n - 1$  variables  $(x_2, \dots, x_n)$ , obtained from  $\mathbf{P}$  by fixing the variable  $x_1$  at the value  $\tilde{x}_1$ . The feasible set of  $\mathbf{P}(\tilde{x}_1)$  is now the convex set  $\mathbf{K}_1 := \mathbf{K} \cap \{\mathbf{x} : x_1 = \tilde{x}_1\}$ . For 0/1 programs where feasibility is easy to detect, one checks whether  $\mathbf{P}(\tilde{x}_1)$  has a feasible solution

and if not, then one sets  $\tilde{x}_1 := 1 - \tilde{x}_1$ . Let  $\mathbf{Y}_2$  be the projection of  $\mathbf{K}_1$  on the  $x_2$ -coordinate axis (or  $\mathbf{Y}_2 = \{0, 1\}$  for 0/1 programs). Then go back to step (b) with now  $x_2 \in \mathbf{Y}_2$  as parameter and  $(x_3, \dots, x_n)$  as variables, to obtain a point  $\tilde{x}_2 \in \mathbf{Y}_2$ , etc.

One ends up with a feasible point  $\tilde{\mathbf{x}} \in \mathbf{K}$ . Then one may use  $\tilde{\mathbf{x}}$  as initial point in a local optimization procedure to find a local minimum  $\hat{\mathbf{x}} \in \mathbf{K}$ . The rationale behind the (J+M)-algorithm is that if  $k$  is large enough and  $\mathbf{P}$  has a unique global minimizer  $\mathbf{x}^* \in \mathbf{K}$  then  $\tilde{\mathbf{x}}$  as well as  $\hat{\mathbf{x}}$  should be close to  $\mathbf{x}^*$ .

The computational cost before the local optimization procedure is less than solving  $n$  times the  $i$ -th semidefinite relaxation in the (J+M)-hierarchy, which is itself the same as the  $i$ -th semidefinite relaxation in the hierarchy defined in [12], with only an additional  $2i$  linear constraints.

#### 10.4.2 The “Joint+Marginal” Algorithm for Continuous Problem

We now assume that the feasible set  $\mathbf{K} \subset \mathbb{R}^n$  of problem  $\mathbf{P}$  is convex and compact. For every  $k \geq 2$ , denote by  $\mathbf{x}_k \in \mathbb{R}^{n-k+1}$  the vector  $(x_k, \dots, x_n)$ , and by  $\tilde{\mathbf{x}}_{k-1} \in \mathbb{R}^{k-1}$  the vector  $(\tilde{x}_1, \dots, \tilde{x}_{k-1})$  (and so  $\tilde{\mathbf{x}}_1 = \tilde{x}_1$ ).

Let  $\mathbf{Y}_1 = [\underline{x}_1, \bar{x}_1] \subset \mathbb{R}$  be the orthogonal projection of  $\mathbf{K}$  on the  $x_1$ -coordinate axis. For every  $\tilde{x}_1 \in \mathbf{Y}_1$ , let the interval  $\mathbf{Y}_2(\tilde{x}_1) \subset \mathbb{R}$  be the orthogonal projection of the set  $\mathbf{K} \cap \{\mathbf{x} : x_1 = \tilde{x}_1\}$  on the  $x_2$ -coordinate axis. Similarly, given  $\tilde{\mathbf{x}}_2 \in \mathbf{Y}_1 \times \mathbf{Y}_2(\tilde{x}_1)$ , let the interval  $\mathbf{Y}_3(\tilde{\mathbf{x}}_2) \subset \mathbb{R}$  be the orthogonal projection of the set  $\mathbf{K} \cap \{\mathbf{x} : x_1 = \tilde{x}_1; x_2 = \tilde{x}_2\}$  on the  $x_3$ -coordinate axis, and etc. in the obvious way.

For every  $k = 2, \dots, n$ , and  $\tilde{\mathbf{x}}_{k-1} \in \mathbf{Y}_1 \times \mathbf{Y}_2(\tilde{x}_1) \cdots \times \mathbf{Y}_{k-1}(\tilde{x}_{k-2})$ , write  $\mathbf{Y}_k(\tilde{\mathbf{x}}_{k-1}) = [\underline{x}_k, \bar{x}_k]$ . Let  $\tilde{f}_k(\mathbf{x}_k) := f((\tilde{\mathbf{x}}_{k-1}, \mathbf{x}_k))$ , and  $\tilde{h}_j^k(\mathbf{x}_k) := h_j((\tilde{\mathbf{x}}_{k-1}, \mathbf{x}_k))$ ,  $j = 1, \dots, m+1$ , where  $\tilde{h}_{m+1}^k(\mathbf{x}_k) = (x_k - \underline{x}_k)(\bar{x}_k - x_k)$ .

Similarly, let  $\mathbf{K}_k(\tilde{\mathbf{x}}_{k-1}) = \{\mathbf{x}_k : (\tilde{\mathbf{x}}_{k-1}, \mathbf{x}_k) \in \mathbf{K}\}$ , i.e.,

$$\mathbf{K}_k(\tilde{\mathbf{x}}_{k-1}) = \{\mathbf{x}_k : \tilde{h}_j^k(\mathbf{x}_k) \geq 0, j = 1, \dots, m+1\}, \quad (10.26)$$

and consider the parametric optimization problem:

$$\mathbf{P}(\tilde{\mathbf{x}}_{k-1}) : \quad J^k(y) = \min \{\tilde{f}_k(\mathbf{x}_k) : \mathbf{x}_k \in \mathbf{K}_k(\tilde{\mathbf{x}}_{k-1}); x_k = y\}, \quad (10.27)$$

obtained from the original problem  $\mathbf{P}$  where the variable  $x_\ell$  is fixed at the value  $\tilde{x}_\ell$ , for every  $\ell = 1, \dots, k-1$ , and the variable  $x_k$  is the parameter. Observe that (10.27) is a particular case of (10.3) where  $\mathbf{x} = \mathbf{x}_k = (x_k, \dots, x_n)$ ,  $\mathbf{y} = x_k$ , and  $\mathbf{Y} = \mathbf{Y}_k = [\underline{x}_k, \bar{x}_k] \neq \emptyset$ .

### 10.4.3 Semidefinite Relaxations

Let  $\varphi_k$  be the probability measure uniformly distributed on  $\mathbf{Y}_k(\tilde{\mathbf{x}}_{k-1})$ , hence with moments  $(\gamma_\ell)$  given by:

$$\gamma_\ell = \int_{\underline{x}_k}^{\bar{x}_k} x^\ell d\varphi_k(x) = \frac{\bar{x}_k^{\ell+1} - \underline{x}_k^{\ell+1}}{(\ell+1)(\bar{x}_k - \underline{x}_k)}, \quad \ell = 0, 1, \dots \quad (10.28)$$

To compute (or at least approximate) the optimal value  $\rho$  of problem (10.5) associated with the parametric optimization problem (10.27), we use the hierarchy of semidefinite relaxations (10.13).

Namely, let  $v_j := \lceil (\deg \tilde{h}_j)/2 \rceil$ ,  $j = 1, \dots, m+1$ , and let  $\mathbf{z}$  be a sequence indexed in the monomial basis of  $\mathbb{R}[\mathbf{x}_k]$ . With index  $i$ , fixed, the parametric semidefinite relaxation (10.13) reads:

$$\begin{aligned} \rho_{ik} &= \inf_{\mathbf{z}} L_{\mathbf{z}}(\tilde{f}_k) \\ \text{s.t. } &\mathbf{M}_i(\mathbf{z}), \mathbf{M}_{i-v_\ell}(\tilde{h}_j^k \mathbf{z}) \geq 0, \quad j = 1, \dots, m+1 \\ &L_{\mathbf{z}}(x_k^\ell) = \gamma_\ell, \quad \ell = 0, 1, \dots, 2i, \end{aligned} \quad (10.29)$$

where  $(\gamma_\ell)$  is defined in (10.28). Its dual is the semidefinite program:

$$\begin{aligned} \rho_{ik}^* &= \sup_{p_i, (\sigma_j)} \int_{\underline{x}_k}^{\bar{x}_k} p_i(x) d\varphi_k(x) \\ \text{s.t. } &f(\mathbf{x}_k) - p_i(x_k) = \sum_{j=0}^{m+1} \sigma_j(\mathbf{x}_k) \tilde{h}_j(\mathbf{x}_k) \\ &p_i \in \mathbb{R}[x_k]_{2i}; \sigma_j \in \Sigma[\mathbf{x}_k], \quad 0 \leq j \leq m+1; \\ &\deg \sigma_j h_j \leq 2i, \quad 0- \leq j \leq m+1. \end{aligned} \quad (10.30)$$

Notice that the size of (10.29) decreases with the index  $k$ . Next, the following result is a direct consequence of Theorem 10.4.

**Theorem 10.6.** *Let  $\mathbf{K}$  be as in (10.24) and assume that  $(h_j)_{j=1}^{m+1}$  satisfy Assumption 10.1. Let  $y \mapsto J^k(y)$  be as in (10.27) and consider the semidefinite relaxations (10.29)-(10.30). Then as  $i \rightarrow \infty$ :*

- (a)  $\rho_{ik} \uparrow \int_{\underline{x}_k}^{\bar{x}_k} J^k(y) d\varphi_k(y)$  and  $\rho_{ik}^* \uparrow \int_{\underline{x}_k}^{\bar{x}_k} J^k(y) d\varphi_k(y)$ .
- (b) Let  $(p_i, (\sigma_j^i))$  be an optimal or nearly optimal solution of (10.30), e.g. such that  $\int_{\underline{x}_k}^{\bar{x}_k} p_i d\varphi_k \geq \rho_{ik}^* - 1/i$ . Then  $p_i(y) \leq J^k(y)$  for all  $y \in \mathbf{Y}_k(\tilde{\mathbf{x}}_{k-1})$ , and

$$\int_{\underline{x}_k}^{\bar{x}_k} |J^k(y) - p_i(y)| d\varphi_k(y) \rightarrow 0, \quad \text{as } i \rightarrow \infty. \quad (10.31)$$

Moreover, if one defines  $\tilde{p}_0 := p_0$ , and  $\tilde{p}_i(y) := \max[\tilde{p}_{i-1}(y), p_i(y)]$ ,  $i = 1, 2, \dots$ , then  $\tilde{p}_i(y) \uparrow J^k(y)$ , for  $\varphi_k$ -almost all  $y \in \mathbf{Y}_k$ , and  $\tilde{p}_i \rightarrow J^k$ ,  $\varphi_k$ -almost uniformly on  $\mathbf{Y}_k$ .

Theorem 10.6 provides a rationale for the (J+M)-algorithm described below.

**(J+M)-algorithm: convex  $\mathbf{K}$  and relaxation index  $i$ :**

Set  $k = 1$ ;

**Step  $k \geq 1$ : Input:** For  $k = 1$ ,  $\tilde{\mathbf{x}}_0 = \emptyset$ ,  $\mathbf{Y}_1(\tilde{\mathbf{x}}_0) = \mathbf{Y}_1$ ;  $\mathbf{P}(\tilde{\mathbf{x}}_0) = \mathbf{P}$ ,  $f_1 = f$  and  $\tilde{h}_j^1 = h_j$ ,  $j = 1, \dots, m$ .

For  $k \geq 2$ ,  $\tilde{\mathbf{x}}_{k-1} \in \mathbf{Y}_1 \times \mathbf{Y}_2(\tilde{\mathbf{x}}_1) \cdots \times \mathbf{Y}_{k-1}(\tilde{\mathbf{x}}_{k-2})$ .

**Output:**  $\tilde{\mathbf{x}}_k = (\tilde{\mathbf{x}}_{k-1}, \tilde{x}_k)$  with  $\tilde{x}_k \in \mathbf{Y}_k(\tilde{\mathbf{x}}_{k-1})$ .

Consider the parametric semidefinite relaxations (10.29) with parameter  $x_k$ , associated with problem  $\mathbf{P}(\tilde{\mathbf{x}}_{k-1})$  in (10.27).

- Extract the univariate polynomial  $p_i \in \mathbb{R}[x_k]_{2i}$ , optimal (or nearly optimal) solution of the dual (10.30).
- Get a global minimizer  $\tilde{x}_k$  of  $p_i$  on the interval  $\mathbf{Y}_k(\tilde{\mathbf{x}}_{k-1}) = [\underline{x}_k, \bar{x}_k]$ , and set  $\tilde{\mathbf{x}}_k := (\tilde{\mathbf{x}}_{k-1}, \tilde{x}_k)$ .

If  $k = n$  stop and output  $\tilde{\mathbf{x}} = (\tilde{x}_1, \dots, \tilde{x}_n) \in \mathbf{K}$ , otherwise set  $k = k + 1$  and repeat.

As  $\mathbf{K}$  is convex,  $\tilde{\mathbf{x}} \in \mathbf{K}$  and one may stop. A refinement is to now use  $\tilde{\mathbf{x}}$  as initial point in a local minimization algorithm to obtain a local minimizer  $\hat{\mathbf{x}} \in \mathbf{K}$  of  $\mathbf{P}$ . In view of Theorem 10.4, the larger the index  $k$  of the relaxations (10.29), the better the values  $f(\tilde{\mathbf{x}})$  and  $f(\hat{\mathbf{x}})$ . Of course, the (J+M)-algorithm can also be used when  $\mathbf{K}$  is not convex. However, it may happen that at some stage  $k$ , the semidefinite relaxation (10.29) may be infeasible because  $J^k(y)$  is infinite for some values of  $y \in \mathbf{Y}_k(\tilde{\mathbf{x}}_{k-1})$ . This is because the feasible set  $\mathbf{K}(\tilde{\mathbf{x}}_{k-1})$  in (10.26) may be disconnected. Moreover, the resulting point  $\tilde{\mathbf{x}}$  is not feasible in general.

### 10.4.3.1 Numerical Experiments

We have tested the (J+M)-algorithm on non convex test problems taken from Floudas et al. [7, Sect. 2]. The set  $\mathbf{K}$  is a convex polytope and the function  $f$  is a nonconvex quadratic polynomial  $\mathbf{x} \mapsto \mathbf{x}'Q\mathbf{x} + \mathbf{b}'\mathbf{x}$  for some real symmetric matrix  $Q$  and vector  $\mathbf{b}$ . We have used the GloptiPoly software.<sup>2</sup> In Table 10.2 below,  $n$  (resp.  $m$ ) stands for the number of variables (resp. constraints), and the value displayed in the “(J+M)-algo” column is obtained in running a local minimization algorithm of

---

<sup>2</sup>The GloptiPoly software [10] is dedicated to solving the generalized problem of moments as described in e.g. [15].

**Table 10.2 (J+M)-algorithm** for convex set  $\mathbf{K}$ 

Prob	$n$	$m$	$f^*$	$i$	(J+M)-algo	Rel. error
2.2	5	11	-17	2	-17.00	0%
2.3	6	8	-361.5	1	-361.50	0%
2.6	10	21	-268.01	1	-267.00	0.3%
2.9	10	21	0	1	0.00	0%
2.8C1	20	30	-394.75	1	-385.30	2.4%
2.8C2	20	30	-884.75	1	-871.52	1.5%
2.8C3	20	30	-8,695	1	-8681.7	0.15%
2.8C4	20	30	-754.75	1	-754.08	0.09%
2.8C5	20	30	-4150.41	1	-3678.2	11%

the MATLAB toolbox with the output  $\tilde{\mathbf{x}}$  of the (J+M)-algorithm as initial guess. As recommended in Glopoly [10] for numerical stability and precision, the problem data have been rescaled to obtain a polytope contained in the box  $[-1, 1]^n$ .

As one may see, and excepted for problem 2.8C5, the relative error is very small. For the last problem the relative error (about 11%) is relatively high despite enforcing some extra upper and lower bounds  $\underline{x}_i \leq x_i \leq \bar{x}_i$ , after reading the optimal solution. However, using  $\tilde{\mathbf{x}} \in \mathbf{K}$  as initial guess of the local minimization algorithm in MATLAB, one still finds the optimal value  $f^*$ .

#### 10.4.4 The “Joint+Marginal” Algorithm for 0/1 Programs

We now consider 0/1 programs, i.e., when  $\mathbf{x} \in \mathbf{K} \cap \{0, 1\}^n$  with  $\mathbf{K}$  as in (10.24), and for problems where finding a feasible point is easy, like e.g., the MAXCUT, knapsack and  $k$ -CLUSTER problems. One may define a 0/1 analogue of the (J+M)-algorithm. In this case  $\mathbf{Y}_k(\tilde{\mathbf{x}}_{k-1}) = \{0, 1\}$  and let  $p \in (0, 1)$  be fixed (e.g.  $p = 1/2$ ). The analogue for 0/1 programs of the parametric-semidefinite relaxation (10.29) reads:

$$\begin{aligned} \rho_{ik} &= \inf_{\mathbf{z}} L_{\mathbf{z}}(\tilde{f}_k) \\ \text{s.t. } \mathbf{M}_i(\mathbf{z}), \mathbf{M}_{i-v_\ell}(\tilde{h}_j^k \mathbf{z}) &\geq 0, \quad j = 1, \dots, m \\ L_{\mathbf{z}}(\mathbf{x}_k^\alpha) &= L_{\mathbf{z}}(\mathbf{x}_k^{I_{\alpha>0}}), \quad \alpha \in \mathbb{N}_{2i}^{n-k} \\ L_{\mathbf{z}}(1) &= 1; L_{\mathbf{z}}(x_k) = p. \end{aligned} \tag{10.32}$$

where  $I_{\alpha>0} = (I_{\alpha_k>0}, \dots, I_{\alpha_n>0})$ , for every  $\alpha \in \mathbb{N}_{2k}^{n-k}$ . The moment constraints  $L_{\mathbf{z}}(\mathbf{x}_k^\alpha) = L_{\mathbf{z}}(\mathbf{x}_k^{I_{\alpha>0}})$  come from the 0/1 constraints  $\mathbf{x}_\ell^2 = x_\ell$  for every  $\ell = k, \dots, n$ . From an optimal solution of the dual of (10.32) one obtains scalar multipliers  $\lambda_0^i$  and  $\lambda_1^i$

associated with the constraints  $L_{\mathbf{z}}(1) = 1$  and  $L_{\mathbf{z}}(x_k) = p$ , from which one forms the univariate affine polynomial  $x_k \mapsto p_i(x_k) := \lambda_0^i + \lambda_1^i x_k$ . Then if  $p_i(0) < p(1)$  one selects  $\tilde{x}_k = 0$  and  $\tilde{x}_k = 1$  otherwise.

To be sure that  $\rho_{ik} < \infty$  (i.e.  $J^k$  is finite on  $\mathbf{Y}_k = \{0, 1\}$ ), one first solves (10.32) without the constraint  $L_{\mathbf{z}}(x_k) = p$ , and with  $\tilde{f}_k = x_k$  (resp.  $-x_k$ ), which yields the optimal value  $\rho_{ik}^0$  (resp.  $\rho_{ik}^1$ ). If  $\rho_{ik}^0 > 0$  (resp.  $\rho_{ik}^1 > -1$ ) then one may set  $x_k = 1$  (resp.  $x_k = 0$ ). And in case where  $\rho_{ik}^0 = 0$  and  $\rho_{ik}^1 = -1$  then  $\rho_{ik} < \infty$ .

For the first step  $k = 1$  of the algorithm, one has:  $\tilde{\mathbf{x}}_0 = \emptyset$ ,  $f_k = f$  and  $\tilde{h}_j^1 = h_j$ ,  $j = 1, \dots, m$ . Then the (J+M)-algorithm for 0/1 programs reads:

**Step k: Input:**  $\tilde{\mathbf{x}}_{k-1} \in \{0, 1\}^{k-1}$ . **Output:**  $\tilde{\mathbf{x}}_k = (\tilde{\mathbf{x}}_{k-1}, \tilde{x}_k) \in \{0, 1\}^k$ .

Consider the parametric semidefinite relaxation (10.32) for problem  $\mathbf{P}(\tilde{\mathbf{x}}_{k-1})$  with parameter  $x_k$ :

- Compute  $\rho_{ik}^0$ ; if  $\rho_{ik}^0 > 0$  then set  $\tilde{x}_k = 1$  else compute  $\rho_{ik}^1$ ; if  $\rho_{ik}^1 > -1$  then set  $\tilde{x}_k = 0$ .
- Else if  $\rho_{ik}^0 = 0$  and  $\rho_{ik}^1 = -1$  compute  $\rho_{ik}$  in (10.32) and extract  $\lambda_0, \lambda_1$  from the dual of (10.32). If  $\lambda_1 < 0$  set  $\tilde{x}_k = 1$ , otherwise set  $\tilde{x}_k = 0$ .
- **Feasibility.**<sup>3</sup> If  $\mathbf{P}((\tilde{\mathbf{x}}_{k-1}, \tilde{x}_k))$  has a feasible solution  $\mathbf{s}_{k+1} \in \{0, 1\}^{n-k}$  then set  $\tilde{\mathbf{x}}_k = (\tilde{\mathbf{x}}_{k-1}, \tilde{x}_k)$ , otherwise set  $\tilde{\mathbf{x}}_k = (\tilde{\mathbf{x}}_{k-1}, 1 - \tilde{x}_k)$ .
- If  $k = n$  then output  $\tilde{\mathbf{x}} \in \{0, 1\}^n$  else  $k := k + 1$  and go to step  $k$ .

Of course, an alternative is standard Branch & Bound. That is, compute the optimal value  $\delta_{0i}$  (resp.  $\delta_{1i}$ ) of  $i$ -th semidefinite relaxation associated with  $\mathbf{P}$  and the additional constraint  $x_1=0$  (resp.  $x_1=1$ ), and decide  $\tilde{x}_1=0$  if  $\delta_{0i} < \delta_{1i}$  (and  $\tilde{x}_1=1$  otherwise); one iterates with  $x_2$  for the 0/1 problem  $\mathbf{P}(\tilde{\mathbf{x}}_1)$ , etc. But this requires to solve two semidefinite relaxations instead of one in the parametric approach. Moreover, the parametric approach can easily deal with *groups* of variables (rather than one variable) at a time. Indeed, with  $s \in \mathbb{N}$  fixed, consider  $(x_1, \dots, x_s) \in \mathbf{Y} := \{0, 1\}^s$  with associated probability distribution  $\varphi$  uniformly distributed on  $\mathbf{Y}$ . Then one now solves the  $i$ -th semidefinite relaxation of the (J+M)-hierarchy applied to problem  $\mathbf{P}$  with  $n-s$  variables  $x_{s+1}, \dots, x_n$  and parameter  $(x_1, \dots, x_s) \in \{0, 1\}^s$ . The dual provides a (square free) polynomial map  $(x_1, \dots, x_s) \mapsto p_i(x_1, \dots, x_s)$  that converges to  $J(x_1, \dots, x_s)$  as  $i$  increases. Then one selects a global minimizer  $\tilde{\mathbf{x}}_s$  of  $p_i$  on  $\mathbf{Y}$  by inspecting  $2^s$  values, and one iterates the procedure with now the 0/1 problem  $\mathbf{P}(\tilde{\mathbf{x}}_s)$  with  $n-s$  variables, etc.

**The max-gap variant.** Recall that in the (J+M)-algorithm, one first computes  $\tilde{x}_1$ , then with  $x_1$  fixed at the value  $\tilde{x}_1$ , one computes  $\tilde{x}_2$ , etc. until one finally computes  $\tilde{x}_n$ , and get  $\tilde{\mathbf{x}}$ . In what we call the “max-gap” variant, one first solves  $n$  programs (10.32) with parameter  $x_1$  to obtain an optimal solution  $p_i(x_1) = \lambda_0^1 + \lambda_1^1 x_1$  of the dual, then with  $x_2$  to obtain  $(\lambda_0^2, \lambda_1^2)$ , etc. finally with  $x_n$  to obtain  $(\lambda_0^n, \lambda_1^n)$ . One then select  $k$  such that  $|\lambda_1^k| = \max_\ell |\lambda_1^\ell|$ , and compute  $\tilde{x}_k$  accordingly. This is because the

---

<sup>3</sup>When  $i$  is large enough, feasibility is guaranteed because  $+\infty > \rho_{ik} \approx \mathbf{P}(\tilde{\mathbf{x}}_k)$  implies that there is a solution  $\mathbf{x}^* \in \mathbf{K} \cap \{0, 1\}^n$  with  $\mathbf{x}_\ell^* = \tilde{x}_\ell$ ,  $\ell = 1, \dots, k$ . For 0/1 problems like MAXCUT,  $k$ -cluster, knapsack, feasibility is easy to check.

**Table 10.3** Relative error for MAXCUT

n	20	30	40
$(\mathbf{P}_1 - \mathbf{Q}_1)/ \mathbf{Q}_1 $	10.3%	12.3%	12.5%

larger  $|\lambda_1|$ , (i.e. the larger  $|p_i(-1) - p_i(1)|$ ), the more likely the choice  $-1$  or  $1$  is correct. After  $x_k$  is fixed at the value  $\tilde{x}_k$ , one repeats the procedure for the  $(n-1)$ -problem  $\mathbf{P}(\tilde{x}_k)$ , etc.

#### 10.4.4.1 Numerical Experiments

We have tested the (J+M)-algorithm on three NP-hard 0/1 problems:

$$\text{MAXCUT : } \min_{\mathbf{x}} \{ \mathbf{x}' Q \mathbf{x} : \mathbf{x} \in \{-1, 1\}^n \} \quad (10.33)$$

$$d-\text{cluster} : \max_{\mathbf{x}} \left\{ \mathbf{x}' Q \mathbf{x} : \mathbf{x} \in \{0, 1\}^n; \sum_{\ell=1}^n x_\ell = d \right\} \quad (10.34)$$

$$0/1-\text{knapsack} : \max_{\mathbf{x}} \left\{ \mathbf{c}' \mathbf{x} : \mathbf{x} \in \{0, 1\}^n; \sum_{\ell=1}^n a_\ell x_\ell \leq b \right\} \quad (10.35)$$

for some real symmetric matrix  $Q = (Q_{ij}) \in \mathbb{R}^{n \times n}$ , vector  $\mathbf{c} \in \mathbb{R}^n$ ,  $\mathbf{a} = (a_\ell) \in \mathbb{N}^n$  and integers  $d, b \in \mathbb{N}$ .

In our sample of randomly generated problems, the entry  $Q_{ij}$  of the real symmetric matrix  $Q$  is set to zero with probability  $1/2$  and when different from zero,  $Q_{ij}$  is randomly (and independently) generated according to the uniform probability distribution on the interval  $[0, 10]$ . Let  $\mathbf{Q}_1$  denote the optimal value of the SDP-relaxation (10.32) without the marginal constraint  $L_{\mathbf{z}}(x_k) = p$ . And so for MAXCUT,  $\mathbf{Q}_1$  is the optimal value of Shor’s relaxation with famous Goemans and Williamson’s 0.878 performance guarantee. Let  $\mathbf{P}_1$  denote the cost of the solution  $\mathbf{x} \in \{-1, 1\}^n$  (or  $\mathbf{x} \in \{0, 1\}^n$ ) generated by the (J+M)-algorithm, and let  $(\mathbf{P}_1 - \mathbf{Q}_1)/|\mathbf{Q}_1|$  be the relative error.

- For the **MAXCUT** problem, we have tested the max-gap variant of the (J+M)-algorithm for problems on random graphs with  $n = 20, 30$  and  $40$  variables, and with  $i = 1$  and  $p = 1/2$ . For each value of  $n$ , we have generated 50 problems and 100 for  $n = 40$ . In (10.13) the parameter  $p \in (0, 1)$  is set to 0.5. In Table 10.3 we have reported the average relative error  $(\mathbf{P}_1 - \mathbf{Q}_1)/|\mathbf{Q}_1|$ , which as one may see, is comparable with the Goemans and Williamson (GW) ratio. In fact, when evaluating the solution with the standard criterion  $\frac{1}{2} \sum_{i < j} Q_{ij}(1 - x_i x_j)$  to maximize, one obtains a relative error of less than 3.5%, very much like for the GW solution obtained after a randomized rounding procedure.
- For the **d-cluster problem** one may add the  $n$  constraints  $x_j(d - \sum_\ell x_\ell) = 0$ ,  $j = 1, \dots, n$ , in the definition (10.34) of  $\mathbf{P}$  because they are redundant. We have tested

**Table 10.4** Relative error for 0/1 knapsack

n	50	60
$(\mathbf{Q}_1 - \mathbf{P}_1)/ \mathbf{Q}_1 $	2.1%	0.62%

the (J+M)-algorithm on problems randomly generated as for MAXCUT, and with  $d = n/2 = 10$ . The average relative error  $(\mathbf{Q}_1 - \mathbf{P}_1)/|\mathbf{Q}_1|$  was:

- 5.7% on four randomly generated problems with  $n = 60$  variables.
- 4.5% and 5.6% on two randomly generated problems with  $n = 70$  variables.  
The “max-gap” variant was a little better ( $\approx 4\%$  and  $\approx 4.5\%$  respectively).
- 5.7% on a problem with  $n = 80$  variables.
- For the **0/1 knapsack problem** one may add the  $n$  redundant constraints  $x_j(b - \sum_\ell a_\ell x_\ell) \geq 0$ , and  $(1 - x_j)(b - \sum_\ell a_\ell x_\ell) \geq 0$ ,  $j = 1, \dots, n$ , in the definition (10.35) of  $\mathbf{P}$ . We have tested the (J+M)-algorithm on a sample of 16 problems with  $n = 50$  variables and 3 problems with  $n = 60$  variables where,  $b = \sum_\ell a_\ell/2$ , and the integers  $a_\ell$ ’s are generated uniformly in  $[10, 100]$ . The vector  $\mathbf{c}$  is generated by:  $c_\ell = s\epsilon + a_\ell$  with  $s = 0.1$  and  $\epsilon$  is a random variable uniformly distributed in  $[0, 1]$ . From the results reported in Table 10.4 one may see that very good relative errors are obtained.

## References

1. Ash, R.B.: Real Analysis and Probability. Academic Press Inc., Boston (1972)
2. Ben-Tal, A., El Ghaoui, L., A. Nemirovski A.: Robustness. In: Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.) Handbook of Semidefinite Programming: Theory, Algorithms, and Applications, pp. 139–162. Kluwer Academic Publishers, Boston (2000)
3. Ben-Tal, A., Boyd, S.A. Nemirovski, A.: Extending Scope of Robust Optimization: Comprehensive Robust Counterparts of Uncertain Problems, Math. Program. Sér. B **107**, 63–89 (2006)
4. Bertsimas, D., Nohadani, O.: Robust optimization for unconstrained simulation-based problems. Oper. Res. **58**, 161–178 (2010)
5. Bertsimas, D., Natarajan, K., Chung-Piaw, T.: Persistence in discrete optimization under data uncertainty. Math. Prog. Ser. B **108**, 251–274 (2006)
6. Bonnans, J.F., Shapiro, A.: Perturbation Analysis of Optimization Problems, Springer, New York (2000)
7. Floudas, C.A., Pardalos, P.M., Adjiman, C., Esposito, W.R., Güümüs, Z.H., Harding, S.T., Klepeis, J.L., Meyer, C.A., Schweiger, C.A.: Handbook of Test Problems in Local and Global optimization. Kluwer Academic Publishers, Dordrecht (1999)
8. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM **42**, 1115–1145 (1995)
9. Henrion D., Lasserre J.B., Savorgnan C.: Approximate volume and integration for basic semialgebraic sets. SIAM Review **51**, 722–743 (2009)
10. Henrion D., Lasserre J.B., Lofberg Y.: Gloptipoly 3: moments, optimization and semidefinite programming. Optim. Methods and Software **24**, 761–779 (2009)
11. Hernández-Lerma, O., Lasserre, J.B.: Discrete-Time Markov Control Processes: Basic Optimality Criteria. Springer, New York (1996)

12. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. *SIAM J. Optim.* **11**, 796–817 (2001)
13. Lasserre, J.B.: Polynomial programming: LP-relaxations also converge. *SIAM J. Optim.* **15**, 383–393 (2004)
14. Lasserre, J.B.: Robust global optimization wih polynomials. *Math. Prog. Ser. B* **107**, 275–293 (2006)
15. Lasserre, J.B.: Moments, Positive Polynomials and Their Applications. Imperial College Press, London (2009)
16. Lasserre, J.B.: A “joint+marginal” approach to parametric polynomial optimization. *SIAM J. Optim.* **20**, 1995–2022 (2010)
17. Natarajan, K., Song, M., Chung-Piaw, T.: Persistency and its applications in choice modelling. *Manag. Sci.* **55**, 453–469 (2009)
18. Putinar, M.: Positive polynomials on compact semi-algebraic sets. *Indiana Univ. Math. J.* **42**, 969–984 (1993)
19. Schweighofer, M.: Optimization of polynomials on compact semialgebraic sets. *SIAM J. Optim.* **15**, 805–825 (2005)

# Chapter 11

## An Introduction to Formally Real Jordan Algebras and Their Applications in Optimization

F. Alizadeh

### 11.1 Introduction

In this chapter we will examine algebraic foundations of optimization problems over *symmetric cones*, sometimes called *self-scaled cones*. Although the subject of convex optimization falls under the umbrella of mathematical analysis and geometry, in some cases, algebraic considerations can greatly simplify both presentation and analysis of problems. Optimization over symmetric cones is the most prominent example, and the algebraic construct underlying such cones are the *formally real Jordan algebras*.

The fundamental role of matrix algebra was already evident when, as a result of the work of the author [6, 8] and Nesterov and Nemirovski [40], semidefinite programming emerged in the early 1990s. Shortly thereafter second order conic programming (SOCP) joined SDP as a focus of attention. At the time it was immediately observed that many techniques and algorithms for semidefinite programming have analogous techniques in SOCP, see Nemirovski and Scheinberg [42] for an early exposition, and Lobo et al. [31] and Alizadeh and Goldfarb [1] for surveys and background information. At that time a question arose as to whether algorithms for semidefinite and second order conic programming can be generalized to convex optimization problems in general. For many interior point methods this question had already been answered affirmatively by the seminal work of Nesterov and Nemirovski [40] with their theory of *self-concordant barrier functions*. However, for the most effective algorithms, that is primal-dual interior point methods, the question was tackled and partially answered by Nesterov and Todd [43, 44], Helmberg et al. [24], Kojima et al. [30], Monteiro [34, 35], Alizadeh et al. [3],

---

F. Alizadeh (✉)

Rutgers Business School and RUTCOR, Rutgers-State University of New Jersey,  
640 Bartholomew Rd, Piscataway, NJ 08854, USA

e-mail: [alizadeh@rutcor.rutgers.edu](mailto:alizadeh@rutcor.rutgers.edu)

and later by many different authors. The work of Nesterov and Todd is especially important here since these authors showed that for certain convex optimization problems (that is conic optimization over what they termed as self-scaled cones) primal-dual interior point methods may be designed which generalize and retain most desirable properties of such methods for linear, semidefinite and second order conic programs. Nesterov and Todd used hardly any algebraic theory, but almost immediately it was recognized that their self-scaled cones are nothing else than *symmetric cones*, which are the cone of squares of formally real Jordan algebras (see below). Güler [19] showed that Nesterov and Nemirovski's self-concordant barriers become especially simple for symmetric cones. Faybusovich in a series of papers [13–15] showed that Jordan algebraic techniques can be used to describe Nesterov–Todd's and some other primal-dual algorithms in a more transparent way. Later Alizadeh and Schmieta, using the machinery of Jordan algebras, showed that the AHO algorithm [47] and in general the entire class of Monteiro–Zhang family of primal-dual methods [34, 58] extend to symmetric cones, [48]. Many more papers have since appeared which use Jordan algebraic techniques to devise algorithms for symmetric conic optimization problems.

Formally real Jordan algebras have become an indispensable topic for researchers who are designing algorithms or developing software for semidefinite and second order conic programming. However, the invention of these algebras initially had nothing to do with optimization. The German physicist Pascual Jordan, who together with Werner Heisenberg, developed the matrix interpretation of quantum mechanics in the 1920s, was looking for a formal theory for the new physics. He started the formalism that is known today as Jordan algebra,<sup>1</sup> but the mathematically sound theory along with its key properties were developed in the paper of Jordan et al. [27]. The quantum physics aspects of Jordan algebras does not seem to have gone too far. However, Jordan algebras have become one of central theories of abstract algebra along with associative, Lie, and alternative algebras, see [49]. There are two versions of Jordan algebras. One is the general theory, and the other is the special case of formally real Jordan algebras. The relation between these two theories shares some essential similarities to the relation between the algebra of general square matrices and the algebra of symmetric (or complex hermitian) matrices: Both theories deal with spectral properties of matrices, and topics such as rank, determinants, characteristic and minimal polynomials, and functions defined on matrices. However, in the special case of real symmetric matrices these topics become especially simple. Jordan algebras generalize these concepts and add some important new notions. Similar to case of square matrices, such theories are simpler on formally real Jordan algebras.

There are many books, from elementary to technical that cover Jordan algebras in general, or formally real Jordan algebras in particular. Two outstanding references are Faraut and Korány [16], and Koecher's Minnesota lecture notes [28]. On general Jordan algebras the text of Jacobson [25] and the newer book [33] should be

---

<sup>1</sup>The Jordan algebra invented by Jordan was actually the formally real version.

mentioned. The text of Springer [51] presents Jordan algebras from a different point of view, namely from that of axiomatizing properties of the inverse operation. Finally, we should mention the small book of Schaefer [49] which is on algebras in general, but it also devotes ample space to Jordan algebras.

In this chapter, we present an introduction to formally real Jordan algebras. We are not going to duplicate what is already ably presented in the references mentioned above. Instead, we plan to introduce and motivate the subject from the point of view of optimization. We hope to convey the point that when working on SDP, SOCP, or related topics, researchers may benefit by taking advantage of the machinery of Jordan algebras. In doing so, at the very least, they may be able to present their results in an elegant and transparent language that simultaneously applies to both SDP and SOCP.

In Sect. 11.2 we demonstrate how familiar concepts in semidefinite and second order conic programming, namely complementarity and barrier functions, naturally lead to algebras. In Sects. 11.3 and 11.4 we introduce the basic properties of formally real Jordan algebras and their most useful properties for optimization purposes. In Sect. 11.5 we examine applications of Jordan algebras to optimization problems. In particular, we discuss non-degeneracy and strict complementarity, and finally show how properties of Jordan algebras make presentation of interior points methods transparent.

## 11.2 From Complementarity and Barriers to Algebras

To see how Jordan algebras arise in optimization problems we start with recalling complementarity relations for certain well-known classes of convex cones. In this chapter we are exclusively concerned with *proper cones* that is those which are closed, convex, pointed, and full-dimensional. Let  $\mathcal{K} \subseteq \mathbb{R}^n$  be a cone, and  $\mathcal{K}^*$  its *dual cone*, that is,

$$\mathcal{K}^* = \{\mathbf{y} \in \mathbb{R}^n \mid \langle \mathbf{x}, \mathbf{y} \rangle \geq 0 \text{ for all } \mathbf{x} \in \mathcal{K}\},$$

where  $\langle \cdot, \cdot \rangle$  is an inner product on  $\mathbb{R}^n$ . The dual cone of a proper cone is always proper. As is well-known, each proper cone induces a partial order  $\succ_{\mathcal{K}}$ , where  $\mathbf{x} \succ_{\mathcal{K}} \mathbf{y}$  means that  $\mathbf{x} - \mathbf{y} \in \mathcal{K}$  (and  $\mathbf{x} \succ_{\mathcal{K}} \mathbf{y}$  if  $\mathbf{x} - \mathbf{y} \in \text{Int } \mathcal{K}$ , the interior of  $\mathcal{K}$ ).

Consider the inequality  $\langle \mathbf{x}, \mathbf{s} \rangle \geq 0$  for  $\mathbf{x} \in \mathcal{K}$  and  $\mathbf{s} \in \mathcal{K}^*$ . Following Hardy, Littlewood and Pólya [23], we can ask what additional conditions should hold to make this into an equality and what are their implications in optimization? The nature of these conditions depend on the cone  $\mathcal{K}$ . For a proper cone  $\mathcal{K}$  and its dual  $\mathcal{K}^*$ , define the *complementarity set*

$$C(\mathcal{K}, \mathcal{K}^*) \stackrel{\text{def}}{=} \{(\mathbf{x}, \mathbf{s}) \mid \mathbf{x} \in \mathcal{K}, \mathbf{s} \in \mathcal{K}^* \text{ and } \langle \mathbf{x}, \mathbf{s} \rangle = 0\}.$$

Then the following result can be shown (see [41] where a proof based on Güler [20] is presented):

**Lemma 11.1.** *For a proper cone  $\mathcal{K}$ , with  $\dim \mathcal{K} = n$ , the set  $C(\mathcal{K}, \mathcal{K}^*)$  is homeomorphic to  $\mathbb{R}^n$ .*

Thus, even though  $C(\mathcal{K}, \mathcal{K}^*)$  is a subset of  $\mathbb{R}^{2n}$ , there are enough restrictions on it to turn it into an only  $n$ -dimensional manifold. Let us examine  $C(\mathcal{K}, \mathcal{K}^*)$  for some familiar cones.

*Example 11.1 (The nonnegative orthant).* When  $\mathcal{K} = \mathcal{L}$  is the nonnegative orthant  $\mathcal{L} = \{\mathbf{x} \mid x_i \geq 0 \text{ for } i = 1, \dots, n\}$ , then under the standard inner product  $\langle \mathbf{x}, \mathbf{s} \rangle = \mathbf{x}^\top \mathbf{s}$ ,  $\mathcal{L}$  is self-dual:  $\mathcal{L}^* = \mathcal{L}$ . In this case if  $\mathbf{x}, \mathbf{s} \in \mathcal{L}$ , they contain only nonnegative components. Thus,  $\langle \mathbf{x}, \mathbf{s} \rangle = 0$  implies  $x_i s_i = 0$  for  $i = 1, \dots, n$ , or in vector notation  $\mathbf{x} \circ \mathbf{s} = \mathbf{0}$ , where “ $\circ$ ” here is the component-wise multiplication of two vectors. This fact of course is the basis of the familiar complementary slackness theorem in linear programming (LP).

*Example 11.2 (Positive semidefinite matrices).* If  $\mathcal{P}$  is the cone of positive semidefinite matrices in any of spaces of real-symmetric, complex hermitian, or quaternion hermitian matrices, then, under the standard inner product  $\langle X, S \rangle \stackrel{\text{def}}{=} \sum_{ij} X_{ij} S_{ij} = \text{Trace } XS$ ,  $\mathcal{P}$  is self-dual. In this case if both  $X$  and  $S$  are  $r \times r$  positive semidefinite matrices, then it is easy to show that the matrix product  $\frac{XS+SX}{2} = 0$ . Again defining a binary operation on matrices as  $X \circ S = \frac{XS+SX}{2}$ , we see that  $X \succcurlyeq_{\mathcal{P}} 0$ ,  $S \succcurlyeq_{\mathcal{P}} 0$ , and  $\langle X, S \rangle = 0$  implies  $X \circ S = 0$ . This is the basis of the complementary slackness theorem in semidefinite programming (SDP). In case of positive semidefinite matrices, we will simply write  $X \succcurlyeq Y$  in place of  $X \succcurlyeq_{\mathcal{P}} Y$ . We should mention that while it may seem that complementarity could also be expressed as  $XS = 0$ , the binary operation  $(X, S) \rightarrow XS$  is not a mapping onto hermitian matrices. The relation  $XS + SX = 0$  is the simplest one which is both bilinear and maps onto hermitian matrices.

Before we study the next example, we introduce the notation, inspired from MATLAB and similar languages, where we attach vectors and matrices, *row-wise* using “;”, and *column-wise* using “:”. Therefore, if  $\mathbf{x}, \mathbf{y}$  are column vectors, then  $(\mathbf{x}^\top, \mathbf{y}^\top)^\top$  is the same vector as  $(\mathbf{x}; \mathbf{y})$ .

*Example 11.3 (Second order cones).* Let  $\mathcal{Q} \in \mathbb{R}^{n+1}$  be the cone defined by all vectors  $\mathbf{x}$  such that  $x_0 \geq \|\bar{\mathbf{x}}\|$ , where  $\mathbf{x} = (x_0; x_1; \dots; x_n)$ ,  $\bar{\mathbf{x}} = (x_1; \dots; x_n)$ , and  $\|\cdot\|$  be the Euclidean norm. This cone is also self-dual under the standard inner product. Recall the Cauchy–Schwarz–Bunyakovsky inequality  $\langle \mathbf{x}, \mathbf{s} \rangle \leq \|\mathbf{x}\| \|\mathbf{s}\|$ , and the fact that equality holds only if  $\mathbf{x}$  and  $\mathbf{s}$  span a line. From this fact it is easy to prove that if  $\mathbf{x}, \mathbf{s} \in \mathcal{Q}$  and  $\langle \mathbf{x}, \mathbf{s} \rangle = 0$  then  $x_0 s_i + x_i s_0 = 0$  for  $i = 1, \dots, n$ . These relations along with  $\langle \mathbf{x}, \mathbf{s} \rangle = 0$  are the basis of the complementary slackness theorem for the second order conic programming (SOCP) problem. Again we can express these relations in vector form by defining the binary operation  $\mathbf{x} \circ \mathbf{s} = (\langle \mathbf{x}, \mathbf{s} \rangle; x_0 \bar{s} + s_0 \bar{\mathbf{x}})$ . Thus, if  $\mathbf{x}, \mathbf{s} \in \mathcal{Q}$  and  $\langle \mathbf{x}, \mathbf{s} \rangle = 0$  then  $\mathbf{x} \circ \mathbf{s} = \mathbf{0}$ .

In all three examples above the complementarity set  $C(\mathcal{K}, \mathcal{K}^*)$  was characterized by a relation  $\mathbf{x} \circ \mathbf{s} = \mathbf{0}$ ; in each case the meaning of the binary operation “ $\circ$ ” is different, but in all cases “ $\circ$ ” is a *bilinear* operation.

The three binary operations above actually have many properties in common, and the algebraic structures they define form a strong foundation unifying many properties of SOCP and SDP, and even linear programming.

We should mention that of course the sets  $C(\mathcal{K}, \mathcal{K}^*)$  for arbitrary proper cones  $\mathcal{K}$  are not always necessarily characterized by bilinear operations. Indeed, Rudolf et al. [41] show that for many common cones bilinear relations alone are not sufficient to characterize  $C(\mathcal{K}, \mathcal{K}^*)$ .

### 11.2.1 Barriers and Interior Point Methods

We now briefly explore one last topic in optimization theory which also leads directly to algebras. Let  $b : \text{Int}\mathcal{K} \rightarrow \mathbb{R}$  be a *barrier function* for a proper cone  $\mathcal{K}$ ; that is,  $b$  is convex, nonnegative, and at least twice differentiable. Furthermore, for any sequence of points  $\mathbf{x}_i \in \text{Int}\mathcal{K}$  converging to a point  $\mathbf{x} \in \text{bd}\mathcal{K}$ , the boundary of  $\mathcal{K}$ , the sequence  $b(\mathbf{x}_i)$  tends to infinity. Let  $b(\mathbf{x})$  be a barrier function for the following mutually dual standard form conic optimization problems:

$$\min\{\langle \mathbf{c}, \mathbf{x} \rangle \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \succ_{\mathcal{K}} \mathbf{0}\} \quad \max\{\mathbf{b}^\top \mathbf{y} \mid A^\top \mathbf{y} + \mathbf{s} = \mathbf{c}, \mathbf{s} \succ_{\mathcal{K}^*} \mathbf{0}\} \quad (11.1)$$

and let  $\mu, \nu > 0$  be two fixed real numbers. It can be shown that the negative of the Legendre transform of  $b(\cdot)$  defined as  $b^*(\mathbf{s}) = \max_{\mathbf{x}}\{\langle \mathbf{x}, \mathbf{s} \rangle - b(\mathbf{x})\}$  is a barrier for  $\mathcal{K}^*$ , see for example Nesterov and Nemirovski [40]. We now replace the minimization and maximization problems above by the following relaxations:

$$\min\{\langle \mathbf{c}, \mathbf{x} \rangle + \mu b(\mathbf{x}) \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \succ_{\mathcal{K}} \mathbf{0}\} \quad (11.2)$$

$$\max\{\mathbf{b}^\top \mathbf{y} - \nu b^*(\mathbf{s}) \mid A^\top \mathbf{y} + \mathbf{s} = \mathbf{c}, \mathbf{s} \succ_{\mathcal{K}^*} \mathbf{0}\} \quad (11.3)$$

If we apply the standard Karush–Kuhn–Tucker conditions to the minimization problem by forming the *Lagrangian function*

$$L(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \langle \mathbf{c}, \mathbf{x} \rangle + \mu b(\mathbf{x}) + \mathbf{y}^\top (\mathbf{b} - A\mathbf{x}),$$

we see that the first order necessary conditions for the optimal solution of the minimization problem in (11.2), say  $\mathbf{x}_\mu$ , imply

$$\begin{aligned} \nabla_{\mathbf{x}} L &= \mathbf{c}^\top - \mu \nabla_{\mathbf{x}} b(\mathbf{x}) - \mathbf{y}^\top A = 0 \\ \nabla_{\mathbf{y}} L &= \mathbf{b}^\top - \mathbf{x}^\top A = 0. \end{aligned}$$

We call the optimal solutions to (11.2)  $\mu$ -optimal. From the fact  $b(\mathbf{x})$  is a convex and differentiable function with gradient  $\nabla_{\mathbf{x}} b(\mathbf{x})$ , it is easy to deduce that the vector  $\mathbf{s}_\mu$

defined as  $\mathbf{s}_\mu = \mu \nabla_{\mathbf{x}} b(\mathbf{x}) \in \text{Int } \mathcal{K}^*$ . Thus, from the optimality condition we see that if  $\mathbf{x}_\mu, \mathbf{y}_\mu$  solve the first order optimality conditions, then  $(\mathbf{y}_\mu, \mathbf{s}_\mu)$  is feasible for Dual, and the quantity  $\langle \mathbf{y}_\mu, \mathbf{s}_\mu \rangle$  is the duality gap. It is possible to apply the same approach to (11.3) and get primal and dual  $v$ -optimal solutions.

The development above is quite general and applies to any cone and any barrier function for that cone. Let us see what results from this development if we apply it to some concrete cones and barriers.

*Example 11.4 (The nonnegative orthant continued).* Let  $\mathcal{K} = \mathcal{L}$  be the nonnegative orthant and let us choose  $b(\mathbf{x}) = -\sum_i \ln(x_i)$  as the barrier function. Then  $\mathbf{s}_\mu = \mu(\frac{1}{x_1}, \dots, \frac{1}{x_n})^\top$ . The relation between  $\mathbf{x}_\mu$  and  $\mathbf{s}_\mu$  can be rewritten as  $x_i s_i = \mu$  for  $i = 1, \dots, n$ , or  $\mathbf{x}_\mu \circ \mathbf{s}_\mu = \mu \mathbf{1}$ , where “ $\circ$ ” is the component-wise multiplication defined in Example 11.1. Note that  $\mathbf{1}$ , the vector composed of all ones, is the identity element of “ $\circ$ ”. Thus, the relation  $\mathbf{x}_\mu \circ \mathbf{s}_\mu = \mu \mathbf{1}$  is a relaxation of the complementary slackness conditions for LP. Indeed, as  $\mu \rightarrow 0$ , then  $\mathbf{x}_\mu \circ \mathbf{s}_\mu \rightarrow \mathbf{0}$ .

*Example 11.5 (The second order cone continued).* If  $\mathcal{Q}$  is the second order cone in  $\mathbb{R}^{n+1}$ , the function  $b(\mathbf{x}) = -\ln(x_0^2 - \sum_{i=1}^n x_i^2) = -\ln(x_0^2 - \|\bar{\mathbf{x}}\|^2)$  is a barrier function. In this case  $\mathbf{s}_\mu = \frac{2\mu}{x_0^2 - \|\bar{\mathbf{x}}\|^2}(x_0; -\bar{\mathbf{x}})$ . Now, inspired by the complementary conditions for  $\mathcal{Q}$  as given in Example 11.3, we see that  $\mathbf{x} \circ \mathbf{s} = \mu \mathbf{e}$  where  $\mathbf{e} = (1; \mathbf{0})$ , the identity element of “ $\circ$ ”. Thus we can accurately call the vector  $\frac{1}{x_0^2 - \|\bar{\mathbf{x}}\|^2}(x_0; -\bar{\mathbf{x}})$  the inverse of  $\mathbf{x}$  whenever  $x_0^2 - \|\bar{\mathbf{x}}\|^2 \neq 0$ . Just like in linear programming the relation  $\mathbf{x} \circ \mathbf{s} = 2\mu \mathbf{e}$  is a relaxed form of the complementary slackness condition.

*Example 11.6 (The positive semidefinite cone continued).* If  $\mathcal{P}$  is the cone of real symmetric (or complex or quaternion hermitian)  $r \times r$  matrices then  $b(X) = -\ln \text{Det}(X)$  is a barrier function. It is not difficult to show that the gradient of this function is given by  $\nabla_X \ln \text{Det}(X) = X^{-1}$ , and thus  $S_\mu = \mu X^{-1}$ . Notice that the inverse with respect to the ordinary matrix multiplication is identical to the inverse with respect to the multiplication given in Example 11.2. As a result the relation  $S_\mu = \mu X_\mu^{-1}$  may be written in the equivalent form  $\frac{X_\mu S_\mu + S_\mu X_\mu}{2} = X_\mu \circ S_\mu = \mu I$ .

The three examples above have certain properties in common. As seen earlier, there is a binary operation “ $\circ$ ” such that complementary slackness for the corresponding conic optimization is expressed as  $\mathbf{x} \circ \mathbf{s} = \mathbf{0}$ . The “ $\circ$ ” operation itself is a bilinear function of  $\mathbf{x}$  and  $\mathbf{s}$ . Moreover, for each of the three cones, there is a barrier function such that application of the Karush–Kuhn–Tucker optimality conditions to the minimization problem results in the following set of equations:

$$\begin{aligned} A\mathbf{x} &= \mathbf{b} \\ A^\top \mathbf{y} + \mathbf{s} &= \mathbf{c} \\ \mathbf{x} \circ \mathbf{s} &= \mu \mathbf{e}, \end{aligned}$$

where  $\mathbf{e}$  is the identity element of the “ $\circ$ ” operation.

This system of equations has the necessary differentiability properties required by the Newton's method. In this context, the Newton's method amounts to replacing  $(\mathbf{x}, \mathbf{y}, \mathbf{s})$  with  $(\mathbf{x} + \Delta\mathbf{x}, \mathbf{y} + \Delta\mathbf{y}, \mathbf{s} + \Delta\mathbf{s})$ , expanding the three equations, and removing all non-linear terms in  $\Delta\mathbf{x}, \Delta\mathbf{y}$  and  $\Delta\mathbf{s}$ . The result is the following system of linear equations:

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^\top & I \\ R_\circ(\mathbf{s}) & 0 & L_\circ(\mathbf{x}) \end{pmatrix} \begin{pmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\mathbf{s} \end{pmatrix} = \begin{pmatrix} \mathbf{b} - A\mathbf{x} \\ \mathbf{c} - A^\top\mathbf{y} - \mathbf{s} \\ \mu\mathbf{e} - \mathbf{x} \circ \mathbf{s} \end{pmatrix} \quad (11.4)$$

Here  $L_\circ(\mathbf{x})$  is the Jacobian of the function  $\mathbf{s} \rightarrow \mathbf{x} \circ \mathbf{s}$  with respect to  $\mathbf{x}$ ; and likewise  $R_\circ(\mathbf{s})$  is the Jacobian of the function  $\mathbf{x} \rightarrow \mathbf{x} \circ \mathbf{s}$  with respect to  $\mathbf{s}$ . Since  $\circ$  is bilinear in  $\mathbf{x}$  and  $\mathbf{s}$ , the entries of  $L_\circ(\mathbf{x})$  and  $R_\circ(\mathbf{s})$  are linear forms in  $\mathbf{x}$  and  $\mathbf{s}$ , respectively. We will see later that  $L_\circ$  and  $R_\circ$  are, respectively, the *left multiplication* and the *right multiplication* operators.

The discussion above leads us directly to certain algebraic structures, the understanding of which will greatly simplify analysis of SDP, SOCP and related optimization problems.

### 11.3 Jordan Algebras and Their Main Properties

We are now ready to focus on Jordan algebras, in particular on formally real Jordan algebras. However, it is useful to study some relevant properties of general algebras, and then arbitrary Jordan algebras first. An *algebra* refers to a *finite dimensional*<sup>2</sup> linear space  $\mathbb{A}$  over the real field  $\mathbb{R}$ , with an additional binary operation, say  $\diamond$ , on its elements. The main requirement is that this binary operation follows distributive laws over addition, or equivalently, if  $\mathbf{a} \diamond \mathbf{b} = \mathbf{c}$ , then components of  $\mathbf{c}$  are bilinear forms in  $\mathbf{a}$  and  $\mathbf{b}$ . Put another way, for each  $\mathbf{a} \in \mathbb{A}$  there is a linear operator  $L_\diamond(\mathbf{a})$ , called the *left multiplication operator*, such that for any  $\mathbf{b} \in \mathbb{A}$  we have  $\mathbf{a} \diamond \mathbf{b} = L_\diamond(\mathbf{a})\mathbf{b}$ . Similarly,  $R_\diamond$ , called the *right multiplication operator*, is defined by  $\mathbf{a} \diamond \mathbf{b} = R_\diamond(\mathbf{b})\mathbf{a}$ . We will also use  $L_\diamond(\mathbf{a})$  and  $R_\diamond(\mathbf{b})$  for the matrix representation of these linear operators.

An algebra is *commutative* if  $\mathbf{a} \diamond \mathbf{b} = \mathbf{b} \diamond \mathbf{a}$ . In this case it is easy to see that  $L_\diamond = R_\diamond$ . In general, the *commutator* operation is defined by  $[\mathbf{a}, \mathbf{b}] = \mathbf{a} \diamond \mathbf{b} - \mathbf{b} \diamond \mathbf{a}$ , and thus, in commutative algebras,  $[\mathbf{a}, \mathbf{b}] = \mathbf{0}$ .

An algebra is *associative* if for any  $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{A}$ , we have  $\mathbf{a} \diamond (\mathbf{b} \diamond \mathbf{c}) = (\mathbf{a} \diamond \mathbf{b}) \diamond \mathbf{c}$ . The most prominent associative algebra is the algebra of square matrices of order  $r$  under the ordinary matrix multiplication, and is denoted by  $\mathbb{M}_r$ . In fact, all associative algebras are *isomorphic* to some *subalgebra* of  $\mathbb{M}_r$ , see for example, [26]. Such an isomorphism is called a *representation* of the associative algebra.

---

<sup>2</sup>In general algebras are defined over finite or infinite dimensional vector spaces, but in this chapter we are exclusively concerned with finite dimensional ones.

An algebra  $(\mathbb{A}, \diamond)$  is *formally real* if, for all  $\mathbf{a} \in \mathbb{A}$ , whenever  $\sum_{i \in I} \mathbf{a}_i \diamond \mathbf{a}_i = \mathbf{0}$ , then each  $\mathbf{a}_i = \mathbf{0}$ . Here the sum is over any finite index set  $I$ .

There are many more concepts in algebras that are needed. In what follows, we explain these concepts as needed in the context of Jordan algebras. For a detailed account on general algebras consult Schaefer [49].

### 11.3.1 Definitions and Basic Properties

**Definition 11.1.** An algebra  $(\mathbb{J}, \circ)$  is a *Jordan algebra* if it is commutative and satisfies the *Jordan identity*: for all  $\mathbf{x}, \mathbf{y} \in \mathbb{J}$ :

$$\mathbf{x}^2 \circ (\mathbf{x} \circ \mathbf{y}) = \mathbf{x} \circ (\mathbf{x}^2 \circ \mathbf{y}),$$

or equivalently  $L(\mathbf{x})$  and  $L(\mathbf{x}^2)$  commute:  $[L(\mathbf{x}), L(\mathbf{x}^2)] = 0$ .

Although Jordan algebras need not have an identity element, it is always possible to affix one to them. In this chapter we are exclusively interested in Jordan algebras with an identity element  $\mathbf{e}$ . Also note that Jordan algebras are not generally associative. Instead, the Jordan identity serves as a weaker relation than the associative law.

Below we list many of key properties of Jordan algebras, including formally real ones. We only sporadically include proofs. The details are ably provided in the classical notes of Koecher [28], and the text of Faraut and Korány [16].

The three algebras associated with linear programming, SOCP and SDP, are in fact examples of Jordan algebras.

*Example 11.7 (The Jordan algebra derived from associative algebras).* Let  $(\mathbb{A}, *)$  be an associative algebra. Define a new operation on elements of  $\mathbb{A}$  as follows:  $\mathbf{a} \circ \mathbf{b} = \frac{\mathbf{a} * \mathbf{b} + \mathbf{b} * \mathbf{a}}{2}$ . Then it is easily seen that  $(\mathbb{A}, \circ)$  is both commutative, and also follows the Jordan identity, and therefore it is a Jordan algebra. It is customary to denote such Jordan algebras by  $\mathbb{A}^+$ . As a concrete example, the set of real  $r \times r$  matrices under ordinary matrix multiplication forms an associative algebra. As a result it induces Jordan multiplication  $A \circ B = \frac{AB+BA}{2}$ . The  $L_\circ$  operator may be expressed as a Kronecker sum:  $L_\circ(X) = \frac{I \otimes X + X \otimes I}{2}$ , see for example [22]. The set of  $r \times r$  complex matrices, and the set of  $r \times r$  quaternionic matrices also form Jordan algebras. However, we must represent these algebras over the field of real numbers. To do so we note that since both complex numbers and quaternions are themselves associative, they have matrix representations of order two (for complex numbers), and of order four (for quaternions). Thus,  $r \times r$  complex matrices can be represented by  $2r \times 2r$  real matrices, and  $r \times r$  quaternionic matrices can be represented by  $4r \times 4r$  real matrices. The Jordan algebra of complex  $r \times r$  matrices may be viewed as a Jordan subalgebra of  $2r \times 2r$  real matrices, and the  $r \times r$  quaternionic matrices form a Jordan subalgebra of  $\mathbb{M}_{4r}$ .

Not all Jordan algebras arise as a subalgebra of some  $\mathbb{A}^+$ , [5]. This fact gives rise to the following definitions:

**Definition 11.2.** A Jordan algebra  $(\mathbb{J}, \circ)$  is called *special* or *representable* if it is isomorphic to a subalgebra of  $\mathbb{A}^+$  for some associative algebra  $\mathbb{A}$ . If no such isomorphism exists, then  $(\mathbb{J}, \circ)$  is called an *exceptional* Jordan algebra.

Since all associative algebras themselves are subalgebras of square matrices, it follows that special Jordan algebras are subalgebras of  $(\mathbb{M}_r, \circ)$  for some integer  $r$ . We will give an example of an exceptional Jordan algebra later.

*Example 11.8 (Jordan algebras of symmetric and hermitian matrices).* The set  $\mathbb{S}_r$  of real symmetric matrices is not closed under ordinary matrix multiplication, but it is closed under “ $\circ$ ”. Thus,  $(\mathbb{S}_r, \circ)$ , the algebra associated with semidefinite programming, is in fact a Jordan algebra. It is also easily seen that this algebra is formally real, since  $A^2$  for a symmetric matrix  $A$  is positive semidefinite, and sum of positive semidefinite matrices cannot equal zero, unless all of them are zero. Likewise, the sets of  $r \times r$  complex hermitian and quaternionic hermitian matrices under “ $\circ$ ” are Jordan algebras. All these algebras are special since they are subalgebras of  $\mathbb{S}_r, \mathbb{S}_{2r}$ , and  $\mathbb{S}_{4r}$ , respectively. Since subalgebras of formally real algebras are necessarily formally real, all these Jordan algebras are formally real.

*Example 11.9 (The spin factor or quadratic algebra).* The algebra associated with SOCP,  $(\mathbb{R}^{n+1}, \circ)$ , is also a Jordan algebra. Commutativity is obvious. To see that Jordan identity holds first observe that for  $\mathbf{x} = (x_0; \bar{\mathbf{x}})$  (where  $\bar{\mathbf{x}} = (x_1; \dots; x_n)$ ), we have

$$L_\circ(\mathbf{x}) = \begin{pmatrix} x_0 & \bar{\mathbf{x}}^\top \\ \bar{\mathbf{x}} & x_0 I \end{pmatrix},$$

which is an arrow shaped matrix. Now,

$$L_\circ(\mathbf{x}^2) = L_\circ((\|\mathbf{x}\|^2; 2x_0 \bar{\mathbf{x}})) = \begin{pmatrix} \|\mathbf{x}\|^2 & 2x_0 \bar{\mathbf{x}}^\top \\ 2x_0 \bar{\mathbf{x}} & \|\mathbf{x}\|^2 I \end{pmatrix}.$$

Then it is a matter of simple inspection to see that  $L_\circ(\mathbf{x}^2)$  and  $L_\circ(\mathbf{x})$  commute. This algebra is also special. In fact, it is induced by the *Clifford algebra*  $C_{2^n}$ , which is a  $2^n$ -dimensional associative algebra, see for example [47] for details and further references.

It may be hard to imagine how one goes about conducting algebraic manipulations to derive new identities when the associative law does not hold. Actually, if we were only concerned with special Jordan algebras  $\mathbb{A}^+$ , which are induced by associative algebras, then there would be no difficulty. However, in the case of exceptional Jordan algebras such an approach is not valid. Indeed, it can be shown that there are identities that are true for special Jordan algebras, but not for exceptional ones, see [25]. Thankfully, there is a convenient technique known as *polarization* which comes to rescue.

**Definition 11.3.** Let  $G(\mathbf{u}_1, \dots, \mathbf{u}_n)$  be a multilinear form on the Jordan algebra  $(\mathbb{J}, \circ)$ . Then the function  $G_p(\mathbf{u}_1, \dots, \mathbf{u}_n) = \sum_{\pi \in S_n} G(\mathbf{u}_{\pi(1)}, \dots, \mathbf{u}_{\pi(n)})$ , where  $S_n$  is the set of all permutations of  $\{1, \dots, n\}$ , is called the *polarized form* of  $G$ .

The importance of polarized form is the following observation. If  $G(\mathbf{u}_1, \dots, \mathbf{u}_n)$  is a multilinear form on a Jordan algebra where its *diagonal*  $g(\mathbf{x}) \stackrel{\text{def}}{=} G(\mathbf{x}, \dots, \mathbf{x}) = \mathbf{0}$ , then  $G_p(\mathbf{u}_1, \dots, \mathbf{u}_n) = \mathbf{0}$ . In fact, we may obtain new identities by expanding  $G(t_1 \mathbf{u}_1 + \dots + t_n \mathbf{u}_n, \dots, t_1 \mathbf{u}_1 + \dots + t_n \mathbf{u}_n) = \mathbf{0}$ , and setting coefficients with respect to each monomial over variables in  $\{t_1, \dots, t_n\}$  to zero. In particular, the coefficient of the monomial  $t_1 \cdots t_n$  is the polarized form of  $G$ .

Let us apply this principle to the Jordan identity  $[L(\mathbf{u}), L(\mathbf{u}^2)] = 0$ . This implies that the diagonal of  $G(\mathbf{u}, \mathbf{v}, \mathbf{w}) = [L(\mathbf{u}), L(\mathbf{v} \circ \mathbf{w})]$  is zero. Thus, we obtain the following relation for its polarized form:

$$[L(\mathbf{u}), L(\mathbf{v} \circ \mathbf{w})] + [L(\mathbf{v}), L(\mathbf{w} \circ \mathbf{u})] + [L(\mathbf{w}), L(\mathbf{u} \circ \mathbf{v})] = 0. \quad (11.5)$$

Setting  $\mathbf{u} = \mathbf{v}$  we get:

$$2[L(\mathbf{u}), L(\mathbf{u} \circ \mathbf{w})] + [L(\mathbf{w}), L(\mathbf{u}^2)] = 0. \quad (11.6)$$

And multiplying by  $\mathbf{v}$ , using commutativity and rearranging so that  $\mathbf{w}$  is on the right of each term we get:

$$L(\mathbf{u}^2 \circ \mathbf{v}) - L(\mathbf{u}^2)L(\mathbf{v}) - 2(L(\mathbf{u} \circ \mathbf{v}) - L(\mathbf{u})L(\mathbf{v}))L(\mathbf{u}) = 0. \quad (11.7)$$

From here, and using induction, one can prove a central fact in Jordan algebras.

**Lemma 11.2.** *Jordan algebras are power-associative, that is, the order of multiplication does not matter in the product  $\mathbf{x} \circ \cdots \circ \mathbf{x}$ . Thus, for all integers  $p, q$ , the matrices  $L(\mathbf{x}^p)$  and  $L(\mathbf{x}^q)$  commute.*

Put another way, the subalgebra generated by a single element  $\mathbf{x}$ , that is the intersection of all Jordan subalgebras of  $(\mathbb{J}, \circ)$  containing  $\mathbf{x}$ , is associative.

In general when two elements  $\mathbf{x}, \mathbf{y} \in \mathbb{J}$  have the property that  $L(\mathbf{x})L(\mathbf{y}) = L(\mathbf{y})L(\mathbf{x})$ , we say that they *operator commute*. In this case for all  $\mathbf{z}$  we have  $\mathbf{x} \circ (\mathbf{y} \circ \mathbf{z}) = \mathbf{y} \circ (\mathbf{x} \circ \mathbf{z})$ . Thus, power-associativity implies that  $\mathbf{x}^p$  and  $\mathbf{x}^q$  operator commute for all nonnegative integers  $p, q$ .

### 11.3.2 The Minimum and Characteristic Polynomials, Eigenvalues, and Inverse Elements

From power-associativity we can build a spectral theory which parallels and indeed generalizes the theory of eigenvalues in matrices. Let  $(\mathbb{J}, \circ)$  be a Jordan algebra with  $\mathbf{e}$  as its identity element. We start with the observation that if for an arbitrary  $\mathbf{x} \in \mathbb{J}$  we

list the powers  $\mathbf{x}^0 \stackrel{\text{def}}{=} \mathbf{e}, \mathbf{x}, \mathbf{x}^2, \mathbf{x}^3, \dots$ , sooner or later we will have a linearly dependent set of vectors, due to finite dimensionality of the underlying linear space. Let  $d$  be the smallest integer such that  $\{\mathbf{e}, \mathbf{x}, \dots, \mathbf{x}^{d-1}\}$  is linearly independent. This means that there are real numbers  $a_1(\mathbf{x}), a_2(\mathbf{x}), \dots, a_d(\mathbf{x})$  such that

$$\mathbf{x}^d - a_1(\mathbf{x})\mathbf{x}^{d-1} + \cdots + (-1)^d a_d(\mathbf{x})\mathbf{e} = \mathbf{0}$$

The polynomial

$$f(t; \mathbf{x}) \stackrel{\text{def}}{=} t^d - a_1(\mathbf{x})t^{d-1} + \cdots + (-1)^d a_d(\mathbf{x})$$

is the *minimum polynomial* of  $\mathbf{x}$ . The integer  $d$  is the *degree* of  $\mathbf{x}$  and is written as  $\deg(\mathbf{x})$ . Finally  $\deg(\mathbb{J}) = \max_{\mathbf{x} \in \mathbb{J}} \deg(\mathbf{x})$ .

Using basic facts from linear algebra, and from the ring of polynomials—especially Gauss's Lemma—one can prove the following theorem [16].

**Theorem 11.1.** *For any integer  $1 \leq d \leq \deg(\mathbb{J})$  the set of degree  $d$  elements are dense in the algebraic variety made up of elements with degree at most  $d$ . Furthermore, for each degree  $d$  element  $\mathbf{y}$  the coefficients  $a_k(\mathbf{y})$  of the minimum polynomial of  $\mathbf{y}$  are homogeneous polynomials of degree  $k$ .*

Let  $\deg(\mathbb{J}) = r$ . An element  $\mathbf{u}$  whose degree  $\deg(\mathbf{u}) = r$  is called a *regular element*. From Theorem 11.1 it follows that the set of regular elements is an open and dense set in  $\mathbb{J}$ . Furthermore, since for these regular elements each  $a_k(\mathbf{u})$  is a homogeneous polynomial, and thus a continuous function on the set of regular elements of  $\mathbb{J}$ , it can be extended to all elements of  $\mathbb{J}$ ; that is  $a_k(\mathbf{u})$  is well-defined for any  $\mathbf{u} \in \mathbb{J}$ . This observation yields:

**Definition 11.4.** Let  $(\mathbb{J}, \circ)$  be a degree- $r$  Jordan algebra. For any element  $\mathbf{x} \in \mathbb{J}$  the polynomial  $F(t; \mathbf{x}) \stackrel{\text{def}}{=} t^r - a_1(\mathbf{x})t^{r-1} + \cdots + a_r(\mathbf{x})$  is the *characteristic polynomial* of  $\mathbf{x}$ . The roots  $\lambda_i$  of the characteristic polynomial are the *eigenvalues* of  $\mathbf{x}$ . The multiplicity of  $\lambda_i$  as a root of  $F(t; \mathbf{x})$  is the *algebraic multiplicity* of  $\lambda_i$ . The multiset of eigenvalues of  $\mathbf{x}$  along with their algebraic multiplicities is the *spectrum* of  $\mathbf{x}$ .

Thus, for regular elements, the characteristic and minimum polynomials coincide. However, for nonregular elements, while the degree of the characteristic polynomial is always  $r$ , the degree of the minimum polynomial is smaller. For example, for the identity element  $\mathbf{e}$ , the minimum polynomial is  $(t - 1)$ , while its characteristic polynomial is  $(t - 1)^r$ .

For every element its minimum polynomial divides its characteristic polynomial. Therefore  $F(\mathbf{x}) = 0$ ; that is the Cayley–Hamilton theorem for matrices extends to Jordan algebras.

The coefficient  $a_1(\mathbf{x})$  is a linear function of  $\mathbf{x}$  and is called the *trace* of  $\mathbf{x}$ ,  $\text{tr}(\mathbf{x}) = a_1(\mathbf{x}) = \sum_{i=1}^r \lambda_i(\mathbf{x})$ ; the coefficient  $a_r(\mathbf{x}) = \prod_{i=1}^r \lambda_i(\mathbf{x})$  is called the *determinant* of  $\mathbf{x}$  and is written  $\det(\mathbf{x})$ . For example, since the sole eigenvalue of  $\mathbf{e}$  is 1 with algebraic multiplicity  $r$ , it follows that  $\text{tr}(\mathbf{e}) = r$  and  $\det(\mathbf{e}) = 1$ .

Suppose  $\det(\mathbf{x}) \neq 0$ . Then the element

$$\mathbf{x}^{-1} \stackrel{\text{def}}{=} \frac{(-1)^r}{\det(\mathbf{x})} (\mathbf{x}^{r-1} - a_1(\mathbf{x})\mathbf{x}^{r-2} + \cdots + (-1)^{r-1} a_{r-1}(\mathbf{x})\mathbf{e}) \quad (11.8)$$

is called the *inverse* of  $\mathbf{x}$ . Thus, only those elements whose determinants are nonzero have an inverse; such elements are called *invertible*. Since the determinant is the product of eigenvalues (including multiplicities), it follows that an element is invertible if and only if it does not have zero as its eigenvalue.

Clearly  $\mathbf{x}^{-1} \circ \mathbf{x} = \mathbf{e}$ . In general, however, there may be many elements  $\mathbf{y}$  such that  $\mathbf{y} \circ \mathbf{x} = \mathbf{e}$ , but only one of these is in the associative subalgebra generated by  $\mathbf{x}$ , and that is  $\mathbf{x}^{-1}$  as defined above. (Recall that in associative algebras if the inverse element, defined as the element  $\mathbf{y}$  such that  $\mathbf{x} * \mathbf{y} = \mathbf{y} * \mathbf{x} = \mathbf{e}$ , exists it is necessarily unique). We now present a few examples.

*Example 11.10 (Eigenvalues of  $(\mathbb{M}_r, \circ)$ ).* For the algebra of square matrices, since the Jordan product is induced by an associative product then the notions of characteristic and minimum polynomials, eigenvalues, trace, determinant and inverse all coincide with the familiar definitions for matrices. The same is true for complex and quaternionic matrices. The degree  $\deg(\mathbb{M}_r) = r$ . And for  $r \times r$  complex matrices and  $r \times r$  quaternion matrices the degree is again  $r$ . Note however, that complex and quaternion matrices have, respectively  $2r \times 2r$  and  $4r \times 4r$  real representations, even if their degree is  $r$ .

*Example 11.11 (Eigenvalues of the spin factor algebra  $(\mathbb{R}^{n+1}, \circ)$ ).* For the spin factor algebra things are considerably simpler. First, it is easily verified that for every  $\mathbf{x} = (x_0; \bar{\mathbf{x}})$  we have

$$\mathbf{x}^2 - 2x_0\mathbf{x} + (x_0^2 - \|\bar{\mathbf{x}}\|^2)\mathbf{e} = \mathbf{0}$$

From this relationship we can immediately conclude that the degree of spin factor algebra is always 2, and in fact, all elements other than multiples of identity  $\mathbf{e}$  are regular. The roots of the characteristic polynomial  $t^2 - 2x_0t + (x_0^2 - \|\bar{\mathbf{x}}\|^2)$  are  $x_0 \pm \|\bar{\mathbf{x}}\|$ , the trace is  $2x_0$  and the determinant is  $\det(\mathbf{x}) = x_0^2 - \|\bar{\mathbf{x}}\|^2$ . The inverse element is given by

$$\mathbf{x}^{-1} = \frac{1}{x_0^2 - \|\bar{\mathbf{x}}\|^2} (x_0, -x_1, \dots, -x_n)^\top \quad (11.9)$$

which is the same as the one we obtained from the gradient of the barrier function  $-\ln(x_0^2 - \|\bar{\mathbf{x}}\|^2) = -\ln \det(\mathbf{x})$  in Example 5.

A bilinear form  $B(\mathbf{x}, \mathbf{y})$  on  $\mathbb{J}$  is *associative* if  $B(\mathbf{x} \circ \mathbf{y}, \mathbf{z}) = B(\mathbf{x}, \mathbf{y} \circ \mathbf{z})$  for all  $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{J}$ . Put another way, for all  $\mathbf{y}$ ,  $B(L(\mathbf{x})\mathbf{y}, \mathbf{z}) = B(\mathbf{y}, L(\mathbf{x})\mathbf{z})$ . Here are two important associative bilinear forms:

**Lemma 11.3.** *The bilinear forms  $T(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \text{Trace } L(\mathbf{x} \circ \mathbf{y})$  and  $\tau(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \text{tr}(\mathbf{x} \circ \mathbf{y})$  are both associative.<sup>3</sup>*

For  $L(\mathbf{x} \circ \mathbf{y})$  the result follows from the fact that  $\text{Trace}(AB) = \text{Trace}(BA)$ . For  $\text{tr}(\mathbf{x} \circ \mathbf{y})$  we may use a form of polarization involving the exponential function; see [16] for details.

The next natural step is to develop the spectral theory—that is the generalization of the notion of eigenvectors—for Jordan algebras. However, we need this theory only for the special case of formally real Jordan algebras, which is simpler than the general theory. We will take up this topic in the next section.

### 11.3.3 The Quadratic Representation

There is another operator which completely characterizes Jordan algebras, and in many contexts is actually more fundamental than the binary operation “ $\circ$ ”. This operator is the generalization of the mapping  $Y \rightarrow XYX$  in matrices. In fact, this operator is expressed by the Kronecker product<sup>4</sup>  $X \otimes X^\top$ . At first glance it seems that this operator depends on ordinary matrix multiplication and it is not clear how to find its generalization to Jordan algebras. However, the following identities suggest a way.

$$\begin{aligned} \frac{XZY + YZX}{2} &= \frac{X \frac{YZ+ZY}{2} + \frac{YZ+ZY}{2} X}{2} + \frac{Y \frac{XZ+ZX}{2} + \frac{XZ+ZX}{2} Y}{2} \\ &\quad - \frac{\frac{XY+YX}{2} Z + Z \frac{XY+YX}{2}}{2} \\ &= X \circ (Y \circ Z) + Y \circ (X \circ Z) - (X \circ Y) \circ Z \end{aligned} \tag{11.10}$$

and thus:

$$XYX = 2X \circ (X \circ Y) - X^2 \circ Y \tag{11.11}$$

These relations can now be extended to all Jordan algebras as follows:

**Definition 11.5.** In a Jordan algebra  $(\mathbb{J}, \circ)$  define:

$$Q_{\mathbf{x}, \mathbf{y}} = L(\mathbf{x})L(\mathbf{y}) + L(\mathbf{y})L(\mathbf{x}) - L(\mathbf{x} \circ \mathbf{y}) \tag{11.12}$$

$$Q_{\mathbf{x}} = Q_{\mathbf{x}, \mathbf{x}} = 2(L(\mathbf{x}))^2 - L(\mathbf{x}^2) \tag{11.13}$$

The operator  $Q_{\mathbf{x}}$  is called the *quadratic representation* of  $\mathbf{x}$ .

<sup>3</sup>We will write  $\text{Trace}(\cdot)$  and  $\text{Det}(\cdot)$  in the usual matrix theoretic sense, and  $\text{tr}(\cdot)$  and  $\det(\cdot)$  for Jordan algebras.

<sup>4</sup>Recall that if  $\text{vec}(X)$  is the vector obtained from stacking columns of the matrix  $X$ , then we have the identity  $\text{vec}(ZYX^\top) = (X \otimes Z)\text{vec}(Y)$ , see for example [21].

*Example 11.12 (The quadratic representation in the spin factor algebra).* For this algebra,

$$Q_x = 2L_o^2(x) - L_o(x^2) = \begin{pmatrix} \|x\|^2 & 2x_0\bar{x}^\top \\ 2x_0\bar{x} & \det(x)I + 2\bar{x}\bar{x}^\top \end{pmatrix} = 2xx^\top - \det(x)R \quad (11.14)$$

where  $\det(x) = x_0^2 - \|\bar{x}\|^2$  and  $R$  is the diagonal reflection matrix with  $R_{00} = 1$  and  $R_{ii} = -1$  for  $i = 1, \dots, n$ .

Some of the most fundamental properties of the quadratic representation are summarized in the following theorem:

**Theorem 11.2 (Properties of  $Q$ ).** *Let  $x$  and  $y$  be elements of a Jordan algebra and let  $x$  be invertible. Then:*

1.  $Q_{x,y} = \frac{1}{2}(Q_{x+y} - Q_x - Q_y)$ .
2.  $Q_{\alpha x} = \alpha^2 Q_x$ .
3.  $Q_x x^{-1} = x$ ,  $Q_x^{-1} = Q_{x^{-1}}$ ,  $Q_{x,y} e = x \circ y$ , and  $Q_x e = x^2$ .
4.  $Q_{x,x^{-1}} Q_x = Q_x Q_{x,x^{-1}} = 2L(x)Q_{e,x} - Q_x = L(x^2)$ .
5.  $Q_{Q_y x} = Q_y Q_x Q_y$ .
6.  $Q_{x^k} = (Q_x)^k$ .
7.  $Q_x L(x^{-1}) = L(x)$ .
8. If  $x, y$  operator commute then  $(Q_x y)^n = Q_{x^n} y^n$ .
9. The gradient  $\nabla_x(\ln \det x) = x^{-1}$  and the Hessian  $\nabla_x^2 \ln \det x = Q_{x^{-1}}$ . In fact, the differential of  $x^{-1}$  in the direction  $u$  is given by  $D_x^u x^{-1} = Q_{x^{-1}} u$ .

We omit proofs here but only mention that some of these identities are proved using polarization. The most fundamental property is (5). Indeed it is possible to start from a proper axiomatization of a linear space along with an operator  $Q_x$ , and recover Jordan product, see [16].

The following property of the quadratic representation will be useful later, and can be proved using polarization.

**Lemma 11.4.** *For any element  $u$  where  $u^2 = e$ , the identity element,  $Q_u$  is an automorphism of the Jordan algebra which is also an involution:  $(Q_u)^2 = I$ .*

## 11.4 Formally Real Jordan Algebras

When a Jordan algebra  $(\mathbb{E}, \circ)$  has the property that it is also formally real, we obtain an additional structure which both enriches and simplifies some of the properties that exist for general Jordan algebras. As we have mentioned before, the analogy between formally real Jordan algebras and the general ones is similar to the analogy between symmetric (or complex hermitian) matrices and general square matrices.

To start, let us define a Jordan algebra  $(\mathbb{E}, \circ)$  to be *Euclidean* if the bilinear form  $T(\mathbf{x}, \mathbf{y}) = \text{Trace } L(\mathbf{x} \circ \mathbf{y})$  is symmetric and positive definite. Recall that for all Jordan algebras  $T(\cdot, \cdot)$  is associative:  $T(L(\mathbf{x})\mathbf{y}, \mathbf{z}) = T(\mathbf{y}, L(\mathbf{x})\mathbf{z})$ . We can now define an inner product  $\langle \mathbf{x}, \mathbf{y} \rangle \stackrel{\text{def}}{=} T(\mathbf{x}, \mathbf{y})$ . Clearly, with respect to this inner product, the matrix representation of the operator  $L(\mathbf{x})$  is necessarily symmetric. This also implies that the quadratic representation  $Q_{\mathbf{x}}$  is representable by a symmetric matrix.

It turns out that Euclidean and formally real Jordan algebras are identical.

**Theorem 11.3.** *The following statements about the Jordan algebra  $(\mathbb{E}, \circ)$  are equivalent.*

- A.  $\mathbb{E}$  is formally real.
- B. The bilinear form  $T(\mathbf{x}, \mathbf{y}) = \text{Trace } L(\mathbf{x} \circ \mathbf{y})$  is symmetric and positive definite.
- C. There exists some positive definite bilinear form  $B(\mathbf{x}, \mathbf{y})$  which is associative.

Koecher in [28] proves this theorem by first showing that all formally real Jordan algebras have only real eigenvalues. From here he proves that statement A implies B. From B to C is trivial and from C to A is also fairly easy.

### 11.4.1 Spectral Properties

Observe that if for any polynomial  $p(t)$  and any element  $\mathbf{x} \in \mathbb{E}$ , if we have  $p(\mathbf{x}) = 0$ , then, by linearity of  $L$ , we have  $L(p(\mathbf{x})) = p(L(\mathbf{x})) = 0$ . Thus,  $f_{\mathbf{x}}(t)$ , the characteristic polynomial of  $\mathbf{x}$  divides the characteristic polynomial of  $L(\mathbf{x})$ . Since  $L(\mathbf{x})$  is symmetric it must have only real eigenvalues, and an orthonormal set of eigenvectors (orthogonality is with respect to  $\langle \cdot, \cdot \rangle$ ). It follows that for formally real Jordan algebras also, all eigenvalues are real. We will start from this observation to build their spectral theory.

To develop the spectral theory, let us first recall a few useful facts from spectral properties of real symmetric matrices. Remember that every real symmetric matrix  $A$  is diagonalizable by an orthogonal matrix  $Q$  with orthonormal columns  $\mathbf{q}_i$ . We order  $A$ 's eigenvalues as follows:  $\lambda_1 > \dots > \lambda_k$ , where each  $\lambda_i$  has a multiplicity of  $m_i$ . Then the spectral decomposition of  $A$  can be written as

$$A = Q\Lambda Q^T \tag{11.15}$$

$$\begin{aligned} &= \lambda_1(\mathbf{q}_{11}\mathbf{q}_{11}^T + \dots + \mathbf{q}_{1m_1}\mathbf{q}_{1m_1}^T) + \dots + \lambda_k(\mathbf{q}_{k1}\mathbf{q}_{k1}^T + \dots + \mathbf{q}_{km_k}\mathbf{q}_{km_k}^T) \\ &= \lambda_1 P_1 + \dots + \lambda_k P_k . \end{aligned} \tag{11.16}$$

where  $\Lambda$  is a diagonal matrix of eigenvalues  $\lambda_i$  of  $A$ , along with their multiplicities. In (11.16)  $P_i = \mathbf{q}_{i1}\mathbf{q}_{i1}^T + \dots + \mathbf{q}_{im_i}\mathbf{q}_{im_i}^T$  are also distinct.

Let us focus on the main properties of matrices  $\mathbf{q}_i \mathbf{q}_i^\top$  and  $P_i$ . They satisfy the following relations:

$$(\mathbf{q}_i \mathbf{q}_i^\top)^2 = \mathbf{q}_i \mathbf{q}_i^\top \quad P_i^2 = P_i \quad (11.17)$$

$$\sum_i \mathbf{q}_i \mathbf{q}_i^\top = I \quad \sum_i P_i = I \quad (11.18)$$

$$(\mathbf{q}_i \mathbf{q}_i^\top)(\mathbf{q}_j \mathbf{q}_j^\top) = (\mathbf{q}_i \mathbf{q}_i^\top) \circ (\mathbf{q}_j \mathbf{q}_j^\top) = 0 \quad P_i P_j = P_i \circ P_j = 0 \text{ whenever } i \neq j \quad (11.19)$$

**Definition 11.6.** An element  $\mathbf{a}$  of an algebra is called an *idempotent* of degree 2 (or simply an idempotent) if  $\mathbf{a}^2 = \mathbf{a}$ .

Thus, the matrices  $\mathbf{q}_i \mathbf{q}_i^\top$  and  $P_i$  are idempotents in both the matrix algebra under ordinary multiplication, and in  $(\mathbb{S}_r, \circ)$ .

Another observation is that the  $P_i$  are in fact in the subalgebra generated by  $A$ , that is each  $P_i$  can be expressed as a polynomial of  $A$ . To see this consider the following system of equations (in  $P_i$ ):

$$\begin{aligned} P_1 + \cdots + P_k &= I \\ \lambda_1 P_1 + \cdots + \lambda_k P_k &= A \\ \vdots \\ \lambda_1^{k-1} P_1 + \cdots + \lambda_k^{k-1} P_k &= A^{k-1} \end{aligned}$$

Since the  $\lambda_i$  are distinct, the matrix of this system is the Vandermonde matrix, which is nonsingular. Using Cramer's rule, each  $P_i$  can be expressed as

$$P_i = \frac{(-1)^i}{V(\lambda_1, \dots, \lambda_k)} \text{Det}_A \begin{pmatrix} 1 & \dots & 1 & I & 1 & \dots & 1 \\ \lambda_1 & \dots & \lambda_{i-1} & A & \lambda_{i+1} & \dots & \lambda_k \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \lambda_1^{k-1} & \dots & \lambda_{i-1}^{k-1} & A^{k-1} & \lambda_{i+1}^{k-1} & \dots & \lambda_k^{k-1} \end{pmatrix}$$

where  $V(\lambda_1, \dots, \lambda_k)$  is the Vandermonde determinant. And with an abuse of notation, by  $\text{Det}_A$  here we mean to treat  $A^k$  as if they are scalars, and expand the “determinant” on the  $i$ th column. With this notation, we see that each  $P_i$  is a polynomial in  $A_i$ .

Finally, for symmetric matrices, the minimum polynomial is simply  $\prod_i^k (t - \lambda_i)$ , while the characteristic polynomial is given by  $\prod_i^k (t - \lambda_i)^{m_i}$ .

Now we are ready to extend the spectral properties of symmetric (or complex hermitian) matrices to formally real Jordan algebras.

**Definition 11.7.** In a formally real Jordan algebra  $(\mathbb{E}, \circ)$ :

1. A set of idempotents  $\mathbf{c}_1, \dots, \mathbf{c}_k$  is called a *complete system of orthogonal idempotents* if

$$\mathbf{c}_1 + \cdots + \mathbf{c}_k = \mathbf{e} \quad \text{the identity element in } \mathbb{E}, \text{ and}$$

$$\mathbf{c}_i \circ \mathbf{c}_j = \mathbf{0} \quad \text{whenever } i \neq j.$$

2. An idempotent  $\mathbf{c}$  is called *primitive* if it cannot be written as sum of two or more nonzero idempotents.
3. A complete system of orthogonal idempotents consisted entirely of primitive idempotents is called a *Jordan frame*.

Thus, if the spectral decomposition of a symmetric matrix  $A$  is given by (11.16) then the  $P_i$  are a complete system of orthogonal idempotents, and  $\mathbf{q}_i \mathbf{q}_i^T$  form a Jordan frame.

*Example 11.13 (Idempotents and Jordan frames in the spin factor algebra).* For the spin factor algebra, as usual, things are much simpler. Let  $\mathbf{q} \in \mathbb{R}^n$  be any vector with  $\|\mathbf{q}\| = 1$ . Then the vectors  $\frac{1}{2}(1, \mathbf{q})$  and  $\frac{1}{2}(1, -\mathbf{q})$  form a Jordan frame. It is easily verified that these are idempotents and their product is zero. In this algebra, the only non-primitive idempotent is the identity element  $\mathbf{e}$ .

Of course, the identity element is an idempotent, but it can be shown that formally real Jordan algebras possess not only idempotents but also Jordan frames, see [25]. We have stated that in formally real Jordan algebras  $L(\mathbf{x})$  is necessarily a symmetric matrix. Now consider for a nonzero element  $\mathbf{x} \in \mathbb{E}$  the subalgebra  $\mathbb{E}(\mathbf{x})$  generated by  $\mathbf{x}$ . This is of course an associative algebra, and also a  $k$ -dimensional subspace of  $\mathbb{E}$ , where  $k = \deg(\mathbf{x})$ . Considering  $L(\mathbf{x})$  as a linear transformation mapping  $\mathbb{E}(\mathbf{x})$  onto itself, (in fact it is a bijection on  $\mathbb{E}(\mathbf{x})$ ), and since  $\dim(\mathbb{E}(\mathbf{x})) = k$ , it follows that  $L(\mathbf{x})$  can be represented by a symmetric  $k \times k$  matrix on  $\mathbb{E}(\mathbf{x})$ . Let us call this matrix  $L_0(\mathbf{x})$ . Let the spectral decomposition of this matrix, following (11.16) be given by

$$L_0(\mathbf{x}) = \lambda_1 P_1 + \cdots + \lambda_k P_k.$$

Also, for each  $P_i$  there is a polynomial  $p_i(t)$  such that  $P_i = p_i(L_0(\mathbf{x}))$ . By associativity of  $\mathbb{E}(\mathbf{x})$  we have  $P_i = L_0(p_i(\mathbf{x}))$ . Now define  $\mathbf{c}_i \stackrel{\text{def}}{=} p_i(\mathbf{x})$ . Then it is easily seen that  $\mathbf{x} = \sum_{i=1}^k \lambda_i \mathbf{c}_i$ , and that the  $\mathbf{c}_i$  form a complete orthogonal system of idempotents in  $\mathbb{E}$ . Thus, the (matrix) spectral decomposition of  $L_0(\mathbf{x})$  leads to the Jordan algebraic *spectral decomposition of  $\mathbf{x}$* .

**Theorem 11.4 (Spectral decomposition version 1).** *For every  $\mathbf{x}$  in the formally real Jordan algebra  $\mathbb{E}$ , there are unique real numbers  $\lambda_1 > \dots > \lambda_k$  and a unique and complete system of orthogonal idempotents  $\mathbf{c}_1, \dots, \mathbf{c}_k$  where  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \cdots + \lambda_k \mathbf{c}_k$ . The  $\lambda_i$  are the set of all eigenvalues of  $\mathbf{x}$ . The minimum polynomial of  $\mathbf{x}$  is  $f(t) \stackrel{\text{def}}{=} \prod_i^k (t - \lambda_i)$ , and  $\deg(\mathbf{x}) = k$ .*

We now turn our attention to Jordan frames and spectral decomposition based on them. Let  $(\mathbb{E}, \circ)$  be a degree  $r$  formally real Jordan algebra. We note that all Jordan frames must have  $r$  elements. This follows from observing that for regular elements, if a Jordan frame has  $s > r$  elements, we can choose  $s$  distinct real numbers  $\lambda_1 > \dots > \lambda_s$ . Then  $\mathbf{x} = \sum_i \lambda_i \mathbf{c}_i$  will have  $s$  distinct eigenvalues, implying that its minimum polynomial will have degree at least  $s$ , contradicting the fact that the degree of  $\mathbb{E}$  is  $r$ . On the other hand a Jordan frame cannot have fewer than  $r$  elements either, since again choosing distinct real numbers  $\lambda_i$  we will have that  $\mathbf{x} = \sum_i \lambda_i \mathbf{c}_i$  is a regular element with fewer than  $r$  eigenvalues, and thus its minimum polynomial has degree less than  $r$ , which is again a contradiction. When  $\mathbf{x}$  is a regular element, then in the spectral decomposition given by Theorem 11.4 the orthonormal idempotents actually form a Jordan frame. Now since the set of regular elements is dense in  $\mathbb{E}$ , every element  $\mathbf{x}$  is a limit point of sequence of regular elements  $\mathbf{x}_i = \lambda_1^{(i)} \mathbf{c}_1^{(i)} + \dots + \lambda_r^{(i)} \mathbf{c}_r^{(i)}$ . It is possible to employ continuity arguments and arrive at the second version of spectral decomposition.

**Theorem 11.5 (Spectral decomposition version 2).** *In any formally real Jordan algebra of degree  $r$ , for every element  $\mathbf{x}$  there are real numbers  $\lambda_{[1]} \geq \lambda_{[2]} \geq \dots \geq \lambda_{[r]}$  and a Jordan frame  $\mathbf{c}_i$  such that  $\mathbf{x} = \lambda_{[1]}\mathbf{c}_1 + \dots + \lambda_{[r]}\mathbf{c}_r$ . The  $\lambda_{[i]}$ 's form exactly the spectrum of  $\mathbf{x}$  (including multiplicities.)*

Note that Jordan frames associated with  $\mathbf{x}$  are not unique if  $\mathbf{x}$  is not regular: for each  $\lambda_i$  of multiplicity  $m_i$ , there can be infinitely many sets of mutually orthogonal primitive idempotents  $\mathbf{c}_1, \dots, \mathbf{c}_{m_i}$ , associated with it. Clearly Theorem 11.5 is a generalization of spectral decomposition of symmetric matrices as given by (11.16).

*Example 11.14 (Spectral decomposition in the spin factor algebra).* For  $\mathbf{x} = (x_0; \bar{\mathbf{x}})$ , define  $\mathbf{c}_1 = \frac{1}{2}(1; \frac{\bar{\mathbf{x}}}{\|\bar{\mathbf{x}}\|})$  and  $\mathbf{c}_2 = \frac{1}{2}(1; -\frac{\bar{\mathbf{x}}}{\|\bar{\mathbf{x}}\|})$ . Then, it is easily seen that  $\mathbf{c}_1, \mathbf{c}_2$  form a Jordan frame, and the spectral decomposition is given by  $\mathbf{x} = (x_0 + \|\bar{\mathbf{x}}\|)\mathbf{c}_1 + (x_0 - \|\bar{\mathbf{x}}\|)\mathbf{c}_2$ .

Let  $\mathbf{c}_1, \mathbf{c}_2$  be idempotents with  $\mathbf{c}_1 \circ \mathbf{c}_2 = \mathbf{0}$ . After plugging them into (11.6) we get that  $[L(\mathbf{c}_1), L(\mathbf{c}_2)] = \mathbf{0}$ . In fact if  $\mathbf{c}_i$  are any complete system of orthogonal idempotents then not only they pairwise commute, but all  $L(\mathbf{c}_i)$  share a common system of eigenvectors. This comes from the fact that if we choose distinct numbers  $\lambda_i$  and set  $\mathbf{x} = \sum_i \lambda_i \mathbf{c}_i$ , then we have seen that each  $\mathbf{c}_i$  is a polynomial in  $\mathbf{x}$ , or equivalently  $\mathbf{c}_i \in \mathbb{E}(\mathbf{x})$ , which is a commutative and associative algebra.

**Lemma 11.5.** *Let  $\mathbf{c}_1, \dots, \mathbf{c}_k$  be a set of mutually orthogonal idempotents. Then any pair of idempotents  $\mathbf{c}_i$  and  $\mathbf{c}_j$  operator commute. Furthermore, all  $L(\mathbf{c}_i)$  share a common set of eigenvectors.*

We also observe that if we fix a Jordan frame  $\mathbf{c}_1, \dots, \mathbf{c}_r$ , then the linear space  $\text{span}\{\mathbf{c}_1, \dots, \mathbf{c}_r\}$  is in fact an associative subalgebra, and all elements of this space operator commute. This is a direct consequence of the fact that  $\mathbf{c}_i$  are a system of complete orthogonal idempotents. In particular, for  $\mathbf{x} = \lambda_{[1]}\mathbf{c}_1 + \dots + \lambda_{[r]}\mathbf{c}_r$ , the inverse, as defined by (11.9), is given by  $\mathbf{x}^{-1} = \lambda_{[1]}^{-1}\mathbf{c}_1 + \dots + \lambda_{[r]}^{-1}\mathbf{c}_r$ , provided all  $\lambda_{[i]}$  are nonzero.

### 11.4.2 The Peirce Decomposition

A fundamental notion in Jordan algebras, and more generally in power-associative algebras, is the notion of *Peirce decomposition*.<sup>5</sup> The idea here is to look at the eigenspaces induced by multiplication operators  $L_c(\cdot)$  and deduce some important properties of the underlying algebra. In this section we focus on the Peirce decomposition of formally real Jordan algebras.

The starting point is a simple observation. Let  $\mathbf{c}$  be an idempotent. Setting  $\mathbf{v} = \mathbf{w} = \mathbf{c}$  in identity (11.7), and noting that  $\mathbf{c}^2 = \mathbf{c}^3 = \mathbf{c}$ , we obtain

$$2L^3(\mathbf{c}) - 3L^2(\mathbf{c}) + L(\mathbf{c}) = 0$$

Thus,  $2t^3 - 3t^2 + t$  is the minimum polynomial of  $L(\mathbf{c})$ . Since this polynomial has the roots 0,  $\frac{1}{2}$  and 1, these are the only eigenvalues of  $L(\mathbf{c})$ , for any idempotent  $\mathbf{c}$ . Since  $L(\mathbf{c})$  is symmetric (with respect to  $\langle \mathbf{x}, \mathbf{y} \rangle = \text{Trace } L(\mathbf{x} \circ \mathbf{y})$ ), the linear space underlying  $\mathbb{E}$  is decomposed into three mutually orthogonal eigenspaces:

$$\mathbb{E}_1(\mathbf{c}) \stackrel{\text{def}}{=} \{\mathbf{x} \in \mathbb{E} \mid L(\mathbf{c})\mathbf{x} = \mathbf{c} \circ \mathbf{x} = \mathbf{x}\} \quad (11.20)$$

$$\mathbb{E}_0(\mathbf{c}) \stackrel{\text{def}}{=} \{\mathbf{x} \in \mathbb{E} \mid L(\mathbf{c})\mathbf{x} = \mathbf{c} \circ \mathbf{x} = \mathbf{0}\} \quad (11.21)$$

$$\mathbb{E}_{\frac{1}{2}}(\mathbf{c}) \stackrel{\text{def}}{=} \left\{ \mathbf{x} \in \mathbb{E} \mid L(\mathbf{c})\mathbf{x} = \mathbf{c} \circ \mathbf{x} = \frac{1}{2}\mathbf{x} \right\}. \quad (11.22)$$

**Definition 11.8.** The eigenspaces  $\mathbb{E}_1(\mathbf{c})$ ,  $\mathbb{E}_0(\mathbf{c})$ , and  $\mathbb{E}_{\frac{1}{2}}(\mathbf{c})$  are the *Peirce spaces* of  $\mathbb{E}$  with respect to  $\mathbf{c}$ . The orthogonal direct sum decomposition of the linear space  $\mathbb{E}$  given by

$$\mathbb{E} = \mathbb{E}_1(\mathbf{c}) \oplus \mathbb{E}_0(\mathbf{c}) \oplus \mathbb{E}_{\frac{1}{2}}(\mathbf{c})$$

is the *Peirce decomposition of  $\mathbb{E}$  with respect to  $\mathbf{c}$* .

Starting from the identity (11.7) and after several algebraic manipulations using properties of idempotents and polarization, we can establish the following theorem:

**Theorem 11.6.** *Let  $\mathbf{c}$  be an idempotent in the formally real Jordan algebra  $\mathbb{E}$ . Then:*

1.  $\mathbb{E}_0(\mathbf{c}) \circ \mathbb{E}_0(\mathbf{c}) = \mathbb{E}_0(\mathbf{c})$  and  $\mathbb{E}_1(\mathbf{c}) \circ \mathbb{E}_1(\mathbf{c}) = \mathbb{E}_1(\mathbf{c})$ , and thus,  $\mathbb{E}_0(\mathbf{c})$  and  $\mathbb{E}_1(\mathbf{c})$  are subalgebras of  $\mathbb{E}$ .
2.  $\mathbb{E}_1(\mathbf{c}) \circ \mathbb{E}_0(\mathbf{c}) = \{\mathbf{0}\}$ .
3.  $\mathbb{E}_{\frac{1}{2}}(\mathbf{c}) \circ \mathbb{E}_1(\mathbf{c}) \subseteq \mathbb{E}_{\frac{1}{2}}(\mathbf{c})$  and  $\mathbb{E}_{\frac{1}{2}}(\mathbf{c}) \circ \mathbb{E}_0(\mathbf{c}) \subseteq \mathbb{E}_{\frac{1}{2}}(\mathbf{c})$ .
4.  $\mathbb{E}_{\frac{1}{2}}(\mathbf{c}) \circ \mathbb{E}_{\frac{1}{2}}(\mathbf{c}) \subseteq \mathbb{E}_0(\mathbf{c}) \oplus \mathbb{E}_1(\mathbf{c})$ .

---

<sup>5</sup>Pronounced like “purse”.

Let  $\mathbf{c}_1$  and  $\mathbf{c}_2$  be two primitive and orthogonal idempotents. If  $\mathbf{w}_0 \in \mathbb{E}_{\frac{1}{2}}(\mathbf{c}_1) \cap \mathbb{E}_{\frac{1}{2}}(\mathbf{c}_2)$  is nonzero, then it is not hard to show that  $\mathbf{w}_0^2 = \frac{1}{2} \|\mathbf{w}_0\| (\mathbf{c}_1 + \mathbf{c}_2)$ , see [16]. In particular if we scale  $\mathbf{w}_0$  such that  $\|\mathbf{w}_0\|^2 = 2$ , and define  $\mathbf{w} = \mathbf{w}_0 + \mathbf{e} - \mathbf{c}_1 - \mathbf{c}_2$ , we can verify that  $\mathbf{w}^2 = \mathbf{e}$ , and thus, by Lemma 11.4,  $Q_{\mathbf{w}}$  is an involutive automorphism of  $(\mathbb{E}, \circ)$ . Furthermore, we have

$$Q_{\mathbf{w}}\mathbf{c}_1 = \mathbf{c}_2$$

Thus we have shown that

**Lemma 11.6.** *If  $\mathbf{c}_1, \mathbf{c}_2$  are orthogonal idempotents, then there is an automorphism of  $(\mathbb{E}, \circ)$  that maps  $\mathbf{c}_1$  to  $\mathbf{c}_2$ .*

It is also useful to explicitly write the orthogonal projection operators on to each of subspaces  $\mathbb{E}_{\lambda}(\mathbf{c})$ .

**Theorem 11.7.** *For each of  $\mathbb{E}_{\lambda}(\mathbf{c})$  with  $\lambda \in \{0, \frac{1}{2}, 1\}$ , and  $\mathbf{c}$  an idempotent, the orthogonal projection operator for  $\mathbf{c}$  is given as follows.*

- $\mathbb{E}_1(\mathbf{c})$ :  $L(\mathbf{c})(2L(\mathbf{c}) - I) = Q_{\mathbf{c}}$ .
- $\mathbb{E}_{\frac{1}{2}}(\mathbf{c})$ :  $4L(\mathbf{c})(I - L(\mathbf{c})) = I - Q_{\mathbf{c}} - Q_{\mathbf{e}-\mathbf{c}}$ .
- $\mathbb{E}_0(\mathbf{c})$ :  $(L(\mathbf{c}) - I)(2L(\mathbf{c}) - I) = Q_{\mathbf{e}-\mathbf{c}}$ .

We also note that if we define  $\mathbf{w} \stackrel{\text{def}}{=} 2\mathbf{c} - \mathbf{e}$  then  $\mathbf{w}^2 = \mathbf{e}$  and by Lemma 11.4,  $Q_{\mathbf{w}}$  is an involutive automorphism of  $\mathbb{E}$ . Also if  $\mathbf{w}^2 = \mathbf{e}$  then it is easy to see that  $\mathbf{c} = \frac{1}{2}(\mathbf{w} + \mathbf{e})$  is an idempotent. Since  $L(\mathbf{c})$  and  $Q_{\mathbf{c}}$  commute, they share the same eigenspaces. Therefore, from the Peirce decomposition, the eigenspaces of  $Q_{\mathbf{c}}$  are  $\mathbb{E}_{\lambda}(\mathbf{c})$  for  $\lambda \in \{0, \frac{1}{2}, 1\}$ .

We are now ready to move to Peirce decomposition with respect to an entire Jordan frame. Let the second version spectral decomposition of  $\mathbf{x}$  be  $\mathbf{x} = \lambda_{[1]}\mathbf{c}_1 + \cdots + \lambda_{[r]}\mathbf{c}_r$ , with  $\mathbf{c}_i$  forming a Jordan frame. Recall from Lemma 11.5 that the  $L(\mathbf{c}_i)$  all share a common system of (necessarily orthogonal) idempotents. Each symmetric matrix  $L(\mathbf{c}_i)$  has only three eigenvalues  $0, \frac{1}{2}$  and  $1$ . If we focus on these common eigenspaces it is not difficult to see that for each  $\mathbf{c}_i$  the space  $\mathbb{E}_1(\mathbf{c}_i)$ , the eigenspace corresponding to eigenvalue  $1$  of  $\mathbf{c}_i$  is also orthogonal to eigenspace  $\mathbb{E}_0(\mathbf{c}_j)$ , the eigenspace of zero of  $L(\mathbf{c}_j)$ , for each  $j \neq i$ . Furthermore, for each pair  $\mathbf{c}_i, \mathbf{c}_j$  with  $i \neq j$  the space  $\mathbb{E}_{\frac{1}{2}}(\mathbf{c}_i) \cap \mathbb{E}_{\frac{1}{2}}(\mathbf{c}_j)$  is also a common eigenspace for all  $\mathbf{c}_k$ .

**Definition 11.9.** Let  $(\mathbb{E}, \circ)$  be a formally real Jordan algebra of degree  $r$ , and let  $\mathbf{c}_1, \dots, \mathbf{c}_r$  be a Jordan frame. Then with respect to this Jordan frame define

$$\mathbb{E}_{ii} \stackrel{\text{def}}{=} \mathbb{E}_1(\mathbf{c}_i) = \{\mathbf{x} \mid \mathbf{c}_i \circ \mathbf{x} = \mathbf{x}\}$$

$$\mathbb{E}_{ij} \stackrel{\text{def}}{=} \mathbb{E}_{\frac{1}{2}}(\mathbf{c}_i) \cap \mathbb{E}_{\frac{1}{2}}(\mathbf{c}_j) = \{\mathbf{x} \mid \mathbf{c}_i \circ \mathbf{x} = \mathbf{c}_j \circ \mathbf{x} = \frac{1}{2}\mathbf{x}\} \quad \text{for } i \neq j$$

Then each  $\mathbb{E}_{ij}$  is a *Peirce space with respect to the Jordan frame  $\mathbf{c}_1, \dots, \mathbf{c}_r$* .

From the properties of eigenspaces we can deduce the following theorem. Its proof is straightforward from Theorem 11.7.

**Theorem 11.8 (Peirce decomposition, version 2 ([16] Theorem IV.2.1)).** *Let  $\mathbb{E}$  be a formally real Jordan algebra, and  $\mathbf{c}_1, \dots, \mathbf{c}_r$  a Jordan frame. Then we have the Peirce decomposition*

$$\mathbb{E} = \bigoplus_{i \leq j} \mathbb{E}_{ij}$$

*The Peirce spaces  $\mathbb{E}_{ij}$  are orthogonal with respect to any symmetric bilinear form.*

This theorem gives a finer decomposition of the space  $\mathbb{E}$ . The most important properties of Peirce decomposition are summarized below:

**Lemma 11.7 (Properties of Peirce spaces).** *Let  $(\mathbb{E}, \circ)$  be a Jordan algebra, and let  $\mathbf{c}_i$ , with  $i = 1, \dots, r$ , form a Jordan frame. Let  $\mathbb{E}_{ij}$  be the Peirce decomposition relative to this frame. If  $i, j, k, l$  are distinct integers between 1 and  $r$ , then:*

1.  $\mathbb{E}_{ii} \circ \mathbb{E}_{ii} \subseteq \mathbb{E}_{ii}$ ,  $\mathbb{E}_{ii} \circ \mathbb{E}_{ij} \subseteq \mathbb{E}_{ij}$ ,  $\mathbb{E}_{ij} \circ \mathbb{E}_{ij} \subseteq \mathbb{E}_{ii} + \mathbb{E}_{jj}$ .
2.  $\mathbb{E}_{ii} \circ \mathbb{E}_{jj} = \{\mathbf{0}\}$ , if  $i \neq j$ .
3.  $\mathbb{E}_{ij} \circ \mathbb{E}_{jk} \subseteq \mathbb{E}_{ik}$ ,  $\mathbb{E}_{ij} \circ \mathbb{E}_{kk} = \{\mathbf{0}\}$ .
4.  $\mathbb{E}_{ij} \circ \mathbb{E}_{kl} = \{\mathbf{0}\}$ .
5.  $\mathbb{E}_0(\mathbf{c}_i) = \bigoplus_{j, k \neq i} \mathbb{E}_{jk}$ .
6. The orthogonal projection to  $\mathbb{E}_{ii}$  is  $Q_{\mathbf{c}_i}$ , and to  $\mathbb{E}_{ij}$  is  $L(\mathbf{c}_i)L(\mathbf{c}_j)$ .

In a formally real Jordan algebra, the Peirce decomposition is closely related to the orthogonal decomposition of the vector space with respect to  $L(\mathbf{x})$ . If  $\mathbf{x} = \sum_{i=1}^k \lambda_i \mathbf{c}_i$  then the Peirce spaces  $\mathbb{E}_{ij}$  corresponding to the system of orthogonal idempotents  $\{\mathbf{c}_1, \dots, \mathbf{c}_k\}$  are eigenspaces of  $L(\mathbf{x})$ . An immediate consequence of this observation is the following:

**Lemma 11.8.** *Let  $\mathbf{x} \in \mathbb{E}$  with spectral decomposition version 1:  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \dots + \lambda_k \mathbf{c}_k$ . Then the following statements hold.*

- i. The matrices,  $L(\mathbf{x})$  and  $\mathbf{Q}_{\mathbf{x}}$  commute and thus share a common system of eigenvectors; in fact the  $\mathbf{c}_i$ 's are among their common eigenvectors.
- ii. The eigenvalues of  $L(\mathbf{x})$  have the form

$$\frac{\lambda_i + \lambda_j}{2} \quad 1 \leq i \leq j \leq k,$$

in particular, all  $\lambda_i$  are eigenvalues of  $L(\mathbf{x})$ .

- iii. The eigenvalues of  $\mathbf{Q}_{\mathbf{x}}$  have the form

$$\lambda_i \lambda_j \quad 1 \leq i \leq j \leq k.$$

**Example 11.15 (Peirce decomposition in  $(\mathbb{S}_n, \circ)$ ).** Let  $\mathbf{q}_i$ , with  $i = 1, \dots, n$ , form an orthonormal basis in  $\mathbb{R}^n$ . Then,  $\mathbf{q}_1 \mathbf{q}_1^\top, \dots, \mathbf{q}_n \mathbf{q}_n^\top$  form a Jordan frame for  $(\mathbb{S}_n, \circ)$ . To see the Peirce spaces associated with the idempotent  $C = \mathbf{q}_1 \mathbf{q}_1^\top + \dots + \mathbf{q}_m \mathbf{q}_m^\top$ , first

observe that  $C$  has eigenvalues 1 (with multiplicity  $m$ ) and 0 (with multiplicity  $n-m$ ). Next recall that  $L(C) = \frac{1}{2}(C \otimes I + I \otimes C)$ , which is half of the Kronecker sum of  $C$  by itself. Also recall that in general the eigenvalues of a Kronecker sum  $A \boxplus B = A \otimes I + I \otimes B^\top$  are pairwise sums of eigenvalues of  $A$  and  $B$ , and the eigenvectors are pairwise Kronecker products of eigenvectors of  $A$  and  $B$ . Likewise, the eigenvalues of the Kronecker product  $A \otimes B$  are pairwise products of eigenvalues  $A$  and  $B$ . In particular,  $A \otimes B$  and  $A \boxplus B$  commute. Thus, since the eigenvalues of  $C$  are 0, and 1, the eigenvalues of  $L(C)$  are  $1, \frac{1}{2}, 0$ . Therefore, the Peirce space  $\mathbb{E}_i(C)$  consists of those matrices  $A \in \mathbb{S}_n$  where  $iA = A \circ (\sum_{i=1}^m \mathbf{q}_i \mathbf{q}_i^\top) = \frac{1}{2} \sum_{i=1}^m (A \mathbf{q}_i \mathbf{q}_i^\top + \mathbf{q}_i \mathbf{q}_i^\top A)$ . It follows that:

$$\begin{aligned}\mathbb{E}_1(C) &= \{A \in \mathbb{S}_n \mid \mathbf{q}_i^\top A \mathbf{q}_j = 0 \quad \text{if } m+1 \leq i \leq n \quad \text{or} \quad m+1 \leq j \leq n\}, \\ \mathbb{E}_0(C) &= \{A \in \mathbb{S}_n \mid \mathbf{q}_i^\top A \mathbf{q}_j = 0 \quad \text{if } 1 \leq i \leq m \quad \text{or} \quad 1 \leq j \leq m\}, \\ \mathbb{E}_{\frac{1}{2}}(C) &= \{A \in \mathbb{S}_n \mid \mathbf{q}_i^\top A \mathbf{q}_j = 0 \quad \text{if } 1 \leq i, j \leq m \quad \text{or} \quad m+1 \leq i, j \leq n\}.\end{aligned}$$

*Example 11.16 (Peirce decomposition in the spin factor algebra).* Let  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \lambda_2 \mathbf{c}_2$  be the spectral decomposition of  $\mathbf{x}$  as given in Example 11.14 with  $\lambda_1 \neq \lambda_2$ . First, it can be verified by inspection that the matrix  $L_o(\mathbf{x})$  has eigenvalues  $\lambda_{1,2} = x_0 \pm \|\bar{\mathbf{x}}\|$ , each with multiplicity 1 and corresponding eigenvectors  $\mathbf{c}_{1,2}$ ; and  $\lambda_3 = x_0$  with multiplicity  $n-1$ . Also,  $\mathbf{Q}_x$ , given in Example 11.12 has eigenvalues  $(x_0 \pm \|\bar{\mathbf{x}}\|)^2$  each with multiplicity one and corresponding eigenvectors  $\mathbf{c}_{1,2}$ . The remaining eigenvalue of  $\mathbf{Q}_x$  is  $\lambda_1 \lambda_2 = \det \mathbf{x} = x_0^2 - \|\bar{\mathbf{x}}\|^2$  with multiplicity  $n-1$ . An idempotent  $\mathbf{c}$  which is not equal to identity element  $\mathbf{e}$  is of the form  $\mathbf{c} = \frac{1}{2}(1, \mathbf{q})$  where  $\mathbf{q}$  is a unit length vector. Thus  $L_o(\mathbf{c})$  has one eigenvalue equal to 1, another equal to 0 and  $n-1$  equal to  $\frac{1}{2}$ . It is easy to verify that

$$\begin{aligned}\mathbb{E}_1(\mathbf{c}) &= \{\alpha \mathbf{c} \mid \alpha \in \mathbb{R}\}, \\ \mathbb{E}_0(\mathbf{c}) &= \{\alpha R \mathbf{c} \mid \alpha \in \mathbb{R}\}, \\ \mathbb{E}_{\frac{1}{2}}(\mathbf{c}) &= \{(0; \mathbf{p}) \mid \mathbf{p}^\top \mathbf{q} = 0\}.\end{aligned}$$

where the matrix  $R$  is the diagonal matrix with  $R_{00} = 1, R_{ii} = -1$ , for  $i = 1, \dots, n$ .

**Definition 11.10.** A formally real Jordan algebra  $\mathbb{E}$  is *simple* if and only if it is not the direct sum algebra of two formally real Jordan subalgebras.

For simple formally real Jordan algebras the following can be shown.

**Theorem 11.9 ([16]).** *If  $\mathbb{E}$  is a formally real Jordan algebra, then it is, in a unique way, a direct sum algebra of simple formally real Jordan algebras.*

In Lemma 11.6 we saw that if  $\mathbf{a}$  and  $\mathbf{b}$  are two orthogonal idempotents, then there is an automorphism of  $\mathbb{E}$  that maps  $\mathbf{a}$  to  $\mathbf{b}$ ; in fact we saw that such an automorphism can be chosen as  $Q_w$  where  $w^2 = \mathbf{e}$ . (Such a  $w$  exists when we have a simple Jordan algebra.) This statement can actually be generalized to two sets of Jordan frames.

**Lemma 11.9.** Let  $\mathbf{c}_1, \dots, \mathbf{c}_r$  and  $\mathbf{d}_1, \dots, \mathbf{d}_r$  be two sets of Jordan frames in the formally real simple Jordan algebra  $(\mathbb{E}, \circ)$ . Then there is an automorphism  $Q$  of  $\mathbb{E}$  such that  $Q\mathbf{c}_i = \mathbf{d}_i$  for  $i = 1, \dots, r$ .

Peirce decomposition helps us establish a relationship between  $r$ , the degree of the formally real Jordan algebra, and  $n$ , its dimension.

**Lemma 11.10 ([16]).** Let  $\mathbb{E}$  be an  $n$ -dimensional simple formally real Jordan algebra and  $(\mathbf{a}, \mathbf{b})$ ,  $(\mathbf{a}_1, \mathbf{b}_1)$  two pairs of orthogonal primitive idempotents. Then there is an integer  $d$  not depending on these idempotents where,

$$\dim\left(\mathbb{E}_{\frac{1}{2}}(\mathbf{a}) \cap \mathbb{E}_{\frac{1}{2}}(\mathbf{b})\right) = \dim\left(\mathbb{E}_{\frac{1}{2}}(\mathbf{a}_1) \cap \mathbb{E}_{\frac{1}{2}}(\mathbf{b}_1)\right) = d, \quad (11.23)$$

and, if  $\text{rk}(\mathbb{E}) = r$ ,  $n = r + \frac{d}{2}r(r - 1)$ .

*Example 11.17 (The parameter  $d$  for  $(\mathbb{S}_n, \circ)$  and  $(\mathbb{R}^{n+1}, \circ)$ ).* For  $(\mathbb{S}_r, \circ)$ ,  $d = 1$ . For  $(\mathbb{R}^{n+1}, \circ)$ ,  $d = n - 1$ .

As a consequence of Lemma 11.8 we can relate the spectrum of  $\mathbf{x}$  and those of  $L(\mathbf{x})$  and  $Q_{\mathbf{x}}$ .

**Lemma 11.11.** Let  $\mathbf{x}, \mathbf{y}$  be two elements of a simple formally real Jordan algebra. Then the following statements are equivalent.

1.  $\mathbf{x}$  and  $\mathbf{y}$  have the same spectrum.
2.  $L(\mathbf{x})$  and  $L(\mathbf{y})$  have the same spectrum.

Furthermore, if  $\mathbf{x}$  and  $\mathbf{y}$  have only positive eigenvalues, then they have the same spectrum if and only if  $Q_{\mathbf{x}}$  and  $Q_{\mathbf{y}}$  have the same spectrum.

The following facts are useful in the analysis of interior point methods for optimization over symmetric cones. Their proofs can be found in [48].

**Proposition 11.1.** Let  $\mathbb{E}$  be the direct sum of simple formally real Jordan algebras  $\mathbb{E}_i$ . Let  $\mathbf{x}, \mathbf{s}, \mathbf{p} \in \mathbb{E}$  have only positive eigenvalues. Define  $\tilde{\mathbf{x}} = \mathbf{Q}_p \mathbf{x}$  and  $\underline{\mathbf{s}} = \mathbf{Q}_{p^{-1}} \mathbf{s}$ . Then:

- i.  $\mathbf{Q}_{\mathbf{x}^{1/2}} \mathbf{s}$  and  $\mathbf{Q}_{\mathbf{s}^{1/2}} \mathbf{x}$  have the same spectrum.
- ii.  $\mathbf{a} = \mathbf{Q}_{\mathbf{x}^{1/2}} \mathbf{s}$  and  $\mathbf{b} = \mathbf{Q}_{\tilde{\mathbf{x}}^{1/2}} \underline{\mathbf{s}}$  have the same spectrum.

Finally we state the classification theorem of formally real Jordan algebras:

**Theorem 11.10 ([16] Chap. V).** Let  $\mathbb{E}$  be a simple formally real Jordan algebra. Then  $\mathbb{E}$  is isomorphic to one of the following algebras:

1. The spin factor algebra  $(\mathbb{R}^{n+1}, \circ)$ .
2. The algebra  $(\mathbb{S}_n, \circ)$  of  $n \times n$  real symmetric matrices.
3. The algebra  $(\mathbb{H}_n, \circ)$  of  $n \times n$  complex hermitian matrices under the operation  $X \circ Y = \frac{1}{2}(XY + YX)$ .
4. The algebra  $(\mathbb{Q}_n, \circ)$  of  $n \times n$  quaternionic hermitian matrices under the operation  $X \circ Y = \frac{1}{2}(XY + YX)$ .

5. The exceptional Albert algebra, that is the algebra  $(\mathbb{O}_3, \circ)$  of  $3 \times 3$  octonion hermitian matrices under the operation,  $X \circ Y = \frac{1}{2}(XY + YX)$ .

Since octonion multiplication is not associative, the 27-dimensional Albert algebra is not induced by an associative operation, as the other four are. This algebra is the only simple, exceptional formally real Jordan algebra.

We should mention that  $\mathbb{O}_2$ , the set of  $2 \times 2$  hermitian octonion matrices also forms a Jordan algebra, but it turns out that this algebra is isomorphic to the spin factor algebra  $(\mathbb{R}^{10}, \circ)$ .

### 11.4.3 Cones of Squares and Their Automorphisms

We are now ready to generalize the notion of positive semidefinite (and positive definite) matrices to formally real Jordan algebras. Among many equivalent characterizations of positive semidefinite (and positive definite) matrices the two that lend themselves best to generalization to formally real Jordan algebras are, (a) the fact that all of their eigenvalues are nonnegative (positive), and (b) the fact that each positive (semi)definite matrix is a square of a unique positive (semi)definite matrix.

**Definition 11.11.** In a formally real Jordan algebra  $(\mathbb{E}, \circ)$  an element  $\mathbf{x}$  is *positive semidefinite* if there is a  $\mathbf{y} \in \mathbb{E}$  such that  $\mathbf{x} = \mathbf{y}^2$ . The set of all positive semidefinite elements of  $\mathbb{E}$  is called the *cone of squares* of  $\mathbb{E}$  and is written as  $\mathcal{K}_{\mathbb{E}}$ .

The fact that the cone of squares is a proper cone can be easily established. First, if  $\mathbf{x} = \mathbf{y}^2$ , then for any  $\alpha \geq 0$ , we have  $\alpha\mathbf{x} = (\sqrt{\alpha}\mathbf{y})^2$ . Second, if the second version spectral decomposition of  $\mathbf{y} = \lambda_{[1]}\mathbf{c}_1 + \dots + \lambda_{[r]}\mathbf{c}_r$ , then  $\mathbf{y}^2 = \lambda_{[1]}^2\mathbf{c}_1 + \dots + \lambda_{[r]}^2\mathbf{c}_r$ ; this follows from orthogonality of  $\mathbf{c}_i$  and the fact that they are idempotents. Thus, the following lemma follows:

**Lemma 11.12.** An element  $\mathbf{x}$  is positive semidefinite if and only if all of its eigenvalues are nonnegative.

Furthermore, to show convexity we first characterize the smallest and the largest eigenvalues.

**Lemma 11.13.** Let  $\mathbf{x} \in \mathbb{E}$  with  $\deg(\mathbf{x}) = k$ . Then we obtain the smallest and the largest eigenvalues of  $\mathbf{x}$  as follows:

$$\lambda_{\min}(\mathbf{x}) = \lambda_k(\mathbf{x}) = \min_{\mathbf{u}} \frac{\langle \mathbf{u}, \mathbf{x} \circ \mathbf{u} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle}, \quad \lambda_{\max}(\mathbf{x}) = \lambda_1(\mathbf{x}) = \max_{\mathbf{u}} \frac{\langle \mathbf{u}, \mathbf{x} \circ \mathbf{u} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle}.$$

*Proof.* Replace  $\mathbf{x} \circ \mathbf{u}$  by  $L(\mathbf{x})\mathbf{u}$  and note that from Lemma 11.8 we get  $\lambda_{\min}(L(\mathbf{x})) = \lambda_{\min}(\mathbf{x})$  and  $\lambda_{\max}(L(\mathbf{x})) = \lambda_{\max}(\mathbf{x})$ .  $\square$

Since all eigenvalues of a square element are nonnegative,  $\mathcal{K}_{\mathbb{E}}$  is pointed. And Lemma 11.13 establishes that  $\mathcal{K}_{\mathbb{E}}$  is convex since if  $\mathbf{x}$  and  $\mathbf{y}$  have nonnegative

eigenvalues, then the smallest eigenvalue of  $\mathbf{x} + \mathbf{y}$  must be nonnegative and at least as large as  $\lambda_{\min}(\mathbf{x}) + \lambda_{\min}(\mathbf{y})$ . We use “ $\geq$ ” for the partial order induced by  $\mathcal{K}_{\mathbb{E}}$ . We call an element *positive definite* (written as  $\mathbf{x} > \mathbf{0}$ ) if all of its eigenvalues are strictly positive.

Now, it is obvious that every idempotent  $\mathbf{c}$  is positive semidefinite, since its eigenvalues are 0 and 1. Since by the second version of spectral decomposition, every element is a linear combination of primitive idempotents, it follows that extreme rays of  $\mathcal{K}_{\mathbb{E}}$  are among primitive idempotents. On the other hand, by definition, primitive idempotents are those which are not sums of other idempotents.

**Lemma 11.14.** *The set of extreme rays of  $\mathcal{K}_{\mathbb{E}}$  are exactly the set of rays generated by primitive idempotents of  $\mathbb{E}$ . Furthermore, the Caratheodory number of  $\mathcal{K}_{\mathbb{E}}$  is  $\deg(\mathbb{E}) = r$ .*

The interior  $\text{Int } \mathcal{K}_{\mathbb{E}}$ , consists of strictly positive definite, and thus invertible, elements of  $\mathcal{K}_{\mathbb{E}}$ .

We now identify the dual cone of  $\mathcal{K}_{\mathbb{E}}$  with respect to inner product  $\langle \mathbf{x}, \mathbf{y} \rangle = \text{Trace } L(\mathbf{x} \circ \mathbf{y})$ . First notice that if  $\mathbf{y} \in \mathcal{K}_{\mathbb{E}}^*$  then for any  $\mathbf{x} = \mathbf{z}^2 \in \mathcal{K}_{\mathbb{E}}$  we have

$$0 \leq \langle \mathbf{x}, \mathbf{y} \rangle = \langle \mathbf{z}^2, \mathbf{y} \rangle = \langle \mathbf{z}, \mathbf{z} \circ \mathbf{y} \rangle = \langle \mathbf{z}, L(\mathbf{y})\mathbf{z} \rangle.$$

Since this is true for any arbitrary  $\mathbf{z}$  it follows that  $L(\mathbf{y})$  is a positive semidefinite matrix, which is equivalent to  $\mathbf{y} \geq \mathbf{0}$ . We have shown that  $\mathcal{K}_{\mathbb{E}}^* \subseteq \mathcal{K}_{\mathbb{E}}$ . Now assume that  $\mathbf{x} \geq \mathbf{0}$ , and let  $\mathbf{y} = \mathbf{z}^2$  be any member of  $\mathcal{K}_{\mathbb{E}}$ . Then

$$\langle \mathbf{y}, \mathbf{x} \rangle = \langle \mathbf{z}^2, \mathbf{x} \rangle = \langle \mathbf{z}, L(\mathbf{x})\mathbf{z} \rangle \geq 0$$

where the last inequality is true since  $L(\mathbf{x}) \geq 0$ . Thus, we have shown that every positive semidefinite element  $\mathbf{x}$  has nonnegative inner product with every other positive semidefinite element, thus  $\mathbf{x} \in \mathcal{K}_{\mathbb{E}}^*$ ; therefore,  $\mathcal{K}_{\mathbb{E}} \subseteq \mathcal{K}_{\mathbb{E}}^*$ . A consequence of Theorem 11.3 is that for simple formally real Jordan algebras all inner products (which are necessarily induced by some associative bilinear form) are related by a constant. As a result we have,

**Lemma 11.15 ([16]).** *The cone of squares  $\mathcal{K}_{\mathbb{E}}$  associated with a formally real Jordan algebra is self-dual with respect to any inner product on  $\mathbb{E}$ .*

We now focus on the automorphism group  $\text{Aut}(\mathcal{K}_{\mathbb{E}})$ , that is the group of linear transformations that map  $\mathcal{K}_{\mathbb{E}}$  back to itself. First, observe that the *dilation group* that is  $(\mathbb{R}_+, \cdot)$ —the set of positive real numbers under multiplication—is a subgroup of  $\text{Aut}(\mathcal{K})$  for any cone  $\mathcal{K}$ .

In the case of cones of squares of formally real Jordan algebras  $\mathbb{E}$ , there are many more symmetries. Let us consider the quadratic representation  $Q_x$ . When we have a special Jordan algebra  $\mathbb{E} = \mathbb{A}^+$  induced by the associative algebra  $(\mathbb{A}, *)$ , we know that  $Q_{xy} = x * y * x$ . In this case if  $x$  is nonsingular then  $y \geq \mathbf{0}$  if and only if  $Q_{xy} \geq \mathbf{0}$ . This is actually true for all formally real Jordan algebras.

**Theorem 11.11.** Let  $\mathbf{x}, \mathbf{y}$  be elements of  $\mathbb{E}$ , a formally real Jordan algebra. And let  $\mathbf{x}$  be invertible. Then  $\mathbf{y} \geq 0$ , if and only if  $\mathbf{Q}_{\mathbf{x}}\mathbf{y} \geq 0$ .

The proof of this theorem as given in [16] uses topological notions. The idea is that the set of all nonsingular elements of  $\mathbb{E}$  is partitioned into connected components; and the component that contains the identity element is  $\text{Int}\mathcal{K}_{\mathbb{E}}$ . Now for a nonsingular element  $\mathbf{x}$ , since the operator  $\mathbf{Q}_{\mathbf{x}}$  is nonsingular, it maps each connected component to another connected component. Since  $\mathbf{Q}_{\mathbf{x}}\mathbf{e} = \mathbf{x}^2$  by Theorem 11.2 part 4 the interior of  $\mathcal{K}_{\mathbb{E}}$  must be mapped to itself. A continuity argument extends the result to all of  $\mathcal{K}_{\mathbb{E}}$ .

Theorem 11.11 establishes that the automorphism group of  $\mathcal{K}_{\mathbb{E}}$  contains all  $\mathbf{Q}_{\mathbf{x}}$  for nonsingular elements  $\mathbf{x}$ . This automorphism group, which is actually a Lie group due to its closedness, acts transitively on the  $\text{Int}\mathcal{K}_{\mathbb{E}}$ . This means that for any two elements  $\mathbf{x}, \mathbf{y} \in \text{Int}\mathcal{K}_{\mathbb{E}}$ , there is an automorphism of  $\mathcal{K}_{\mathbb{E}}$  that maps  $\mathbf{x}$  to  $\mathbf{y}$ . To see this note that for each  $\mathbf{x} \in \text{Int}\mathcal{K}_{\mathbb{E}}$  we have

$$\mathbf{Q}_{\mathbf{x}^{-1}}\mathbf{x}^2 = \mathbf{Q}_{\mathbf{x}^{-1}}L(\mathbf{x})\mathbf{x} = L(\mathbf{x})\mathbf{Q}_{\mathbf{x}^{-1}}\mathbf{x} = L(\mathbf{x})\mathbf{x}^{-1} = \mathbf{e}.$$

where the second equality is due to fact that  $\mathbf{Q}_{\mathbf{x}^{-1}} = (\mathbf{Q}_{\mathbf{x}})^{-1}$  and  $L(\mathbf{x})$  commute, and the third equality comes from Theorem 11.2 part 4. Thus we have shown that every element  $\mathbf{x}^2$  in the interior of the cone of squares can be mapped to the identity element by an automorphism. Conversely,  $\mathbf{Q}_{\mathbf{x}}\mathbf{e} = \mathbf{x}^2$  shows that the identity element can be mapped to any element  $\mathbf{x}^2$  of the cone of squares. Thus, we have shown that the automorphism group acts transitively on the interior of the cone of squares. In fact we can be a bit more precise. Lemma 11.9 shows that the automorphism group acts transitively on Jordan frames. This fact turns out to be critical in some applications in optimization over symmetric cones. In general if the automorphism group of a cone  $\mathcal{K}$  acts transitively on its interior, then  $\mathcal{K}$  is called a *homogeneous cone*.

**Definition 11.12.** A proper cone is *symmetric* if it is self-dual and homogeneous.

We have shown that the cone of squares of a formally real Jordan algebra is symmetric. It can also be shown that any symmetric cone is the cone of squares of some formally real Jordan algebra, see [53].

In general, the automorphism group contains more than one connected component, the one that contains the identity matrix also acts transitively on  $\text{Int}\mathcal{K}_{\mathbb{E}}$ ; we denote this subgroup by  $G_{\mathbb{E}}$ .

As mentioned earlier, a subgroup of  $\text{Aut}(\mathcal{K}_{\mathbb{E}})$  is the dilation group. For mapping Jordan frames to Jordan frames, dilations are not necessary. A subgroup of automorphisms that maps Jordan frames to Jordan frames is  $K_{\mathbb{E}} = G_{\mathbb{E}} \cap O_n$ , where  $O_n$  is the group of orthogonal matrices.

*Example 11.18 (Automorphism group of positive semidefinite real matrices).* For the cone of positive semidefinite matrices  $\mathcal{P}_n$ , observe that  $X$  is positive definite if and only if  $PXP^T$  is positive definite for all nonsingular matrices  $P$ . Thus the set

$\text{Aut}(\mathbb{S}) = \{P \otimes P \mid P \in GL_n\}$  is a subset of  $\text{Aut}(\mathcal{P}_n)$ ; in fact this group is isomorphic to  $GL_n$ , the group of nonsingular  $n \times n$  matrices. Recall that  $SO_n$  is the group of orthogonal matrices with determinant 1; it is the connected component of  $O_n$  containing the identity matrix. Let  $Q \in SO_n$ . The way  $Q$  acts on a positive semidefinite matrix  $X$  is by the operation  $X \rightarrow QXQ^\top$ . In other words, in the space of symmetric matrices,  $K_{\mathcal{P}} = \{Q \otimes Q \mid Q \in SO_n\}$ , which is isomorphic to  $SO_n$ . Thus  $K_{\mathbb{S}_n} = SO_n$ . This group acts transitively on the Jordan frames of  $\mathbb{S}_n$ , that is it maps orthogonal matrices to orthogonal matrices.

*Example 11.19 (Automorphism group of the second order cone).* For the second order cone  $Q \subseteq \mathbb{R}^{n+1}$ , the automorphism group is  $SO_{1,n} \oplus \mathbb{R}_+$ , and  $K = SO_n$ . The group  $SO_{1,n}$  consists of matrices that preserve the bilinear form  $x_0y_0 - \langle \bar{\mathbf{x}}, \bar{\mathbf{y}} \rangle$ ; it includes hyperbolic rotations in any two dimensional plane spanned by a Jordan frame  $\{\mathbf{c}_1, \mathbf{c}_2\}$ . The set of transformations fixing the identity element  $\mathbf{e} = (1; \mathbf{0})$ , that is  $K_Q$ , consists entirely of all orthogonal transformations of  $\mathbb{R}^n$  with determinant 1.

Finally, we call a symmetric cone *simple* or *indecomposable* if the underlying Jordan algebra is simple. Since by Theorem 11.10 there are only five classes simple Jordan algebras, it follows that there are only five classes of simple symmetric cones.

1. The class of second order cones  $Q_{n+1} = \{\mathbf{x} \in \mathbb{R}^{n+1} \mid x_0 \geq \|\bar{\mathbf{x}}\|\}$ .
2. The class of real symmetric positive semidefinite matrices  $\mathcal{P}_n = \{X \in \mathbb{S}_n \mid X \geq 0\}$ .
3. The class of positive semidefinite complex hermitian matrices  $\mathcal{P}_n^{\mathbb{C}} = \{X \in \mathbb{H}_r \mid X \geq 0\}$ .
4. The class of quaternionic hermitian positive semidefinite matrices  $\mathcal{P}_n^{\mathbb{Q}} = \{X \in \mathbb{Q}_n \mid X \geq 0\}$ .
5. The exceptional 27 dimensional cone of  $3 \times 3$  hermitian matrices over octonions which are also positive semidefinite,  $\mathcal{P}_3^{\mathbb{O}} = \{X \in \mathbb{O}_3 \mid X \geq 0\}$ .

#### 11.4.4 Functions Defined on Jordan Algebras and Spectral Functions

Recall that two elements  $\mathbf{x}, \mathbf{y}$  of a formally real Jordan algebra operator commute if  $L(\mathbf{x})$  and  $L(\mathbf{y})$  commute. In [48] using Peirce decomposition the following statement is proved.

**Theorem 11.12 ([48] Theorem 27).** *Two elements  $\mathbf{x}$  and  $\mathbf{y}$  operator commute if and only if they share a common Jordan frame.*

The existence of spectral decomposition in formally real Jordan algebras opens an entire world in which many objects defined for real numbers may be extended to vectors in a formally real Jordan algebra. We start with real valued functions  $f(\cdot)$  defined on  $D \subseteq \mathbb{R}$ . Then one can extend  $f(\cdot)$  to formally real Jordan algebras as follows:

**Definition 11.13.** Let the (version 1) spectral decomposition of  $\mathbf{x}$  is  $\lambda_1 \mathbf{c}_1 + \cdots + \lambda_k \mathbf{c}_k$ . If  $f(\cdot)$  is a continuous real-valued function with a domain  $D \subseteq \mathbb{R}$ , then the extension of  $f$  to  $\mathbb{E}$  is:

$$f(\mathbf{x}) \stackrel{\text{def}}{=} f(\lambda_1)\mathbf{c}_1 + \cdots + f(\lambda_k)\mathbf{c}_k.$$

provided that all  $\lambda_i \in D$ .

This definition is given only for real analytic functions by many authors. However, there is no reason to restrict ourselves in this case.

Here are some of the most important functions on Jordan algebras. Assume the spectral decomposition of  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \cdots + \lambda_r \mathbf{c}_r$ .

1. **The exponential function:**  $\exp(\mathbf{x}) \stackrel{\text{def}}{=} \exp(\lambda_1)\mathbf{c}_1 + \cdots + \exp(\lambda_r)\mathbf{c}_r$ .
2. **Power functions:** For any real number  $t$  and any  $\mathbf{x} > 0$ ,  $\mathbf{x}^t = \lambda_1^t \mathbf{c}_1 + \cdots + \lambda_k^t \mathbf{c}_k$ .

In addition, we can define various norms on elements of formally real Jordan algebras.

**Definition 11.14.** Let the (version 2) spectral decomposition of  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \cdots + \lambda_r \mathbf{c}_r$ , where  $\mathbf{c}_1, \dots, \mathbf{c}_r$  is a Jordan frame.

1. The *Frobenius norm* is  $\|\mathbf{x}\|_F \stackrel{\text{def}}{=} (\lambda_1^2 + \cdots + \lambda_r^2)^{1/2}$ .
2. The *spectral norm* is  $\|\mathbf{x}\| = \max_i\{|\lambda_i|\}$ .
3. The  $l_p$  norm for  $1 \leq p \leq \infty$  is  $\|\mathbf{x}\|_p \stackrel{\text{def}}{=} (|\lambda_1|^p + \cdots + |\lambda_r|^p)^{1/p}$ . In particular  $\|\mathbf{x}\|_1 = |\lambda_1| + \cdots + |\lambda_r|$ , and  $\|\mathbf{x}\|_\infty = \|\mathbf{x}\|$ , the spectral norm.

#### 11.4.4.1 Spectral Functions

A set  $S \subseteq \mathbb{E}$  is called *symmetric* if for every permutation matrix  $P$  (representing a permutation  $\pi$  of  $\{1, \dots, r\}$ ), we have  $P(S) = \{P\mathbf{x} \mid \mathbf{x} \in S\} \subseteq S$ . Similarly, a function  $f : S \rightarrow \mathbb{R}$  defined on a symmetric set  $S$  is called a *symmetric function* if  $f(P\mathbf{x}) = f(\mathbf{x})$  for all permutation matrices  $P$ , and all  $\mathbf{x} \in S$ . For a symmetric set  $S$  we say the set  $\{a_1, \dots, a_r\} \in S$  when at least one (and thus all) vectors  $(a_{\pi_1}; \dots; a_{\pi_r}) \in S$ .

Let  $\mathbf{x} \in \mathbb{E}$  have the version 2 spectral decomposition  $\mathbf{x} = \lambda_{[1]}\mathbf{c}_1 + \cdots + \lambda_{[r]}\mathbf{c}_r$ . Let  $f$  be a symmetric function defined on  $S$ , and  $\{\lambda_{[1]}(\mathbf{x}), \dots, \lambda_{[r]}(\mathbf{x})\} \in S$ . Then  $g(\mathbf{x}) = f(\lambda_{[\pi_1]}(\mathbf{x}), \dots, \lambda_{[\pi_r]}(\mathbf{x}))$  defines a *spectral function* of  $\mathbf{x}$ . The  $l_p$  norms defined above are all examples of spectral functions. Let  $\lambda_{[i]}(\mathbf{x})$  be the  $i^{\text{th}}$  absolute-value-wise largest eigenvalue of  $\mathbf{x}$ :  $|\lambda_{[1]}| \geq \cdots \geq |\lambda_{[r]}|$ . Also, let  $\lambda(\mathbf{x}) = (\lambda_{[1]}(\mathbf{x}), \dots, \lambda_{[r]}(\mathbf{x}))$ . Then the following are some of the most common spectral functions:

1.  $\max_k(\lambda(\mathbf{x})) = \sum_{i=1}^k \lambda_{[i]}(\mathbf{x})$ , and  $\min_k(\lambda(\mathbf{x})) \sum_{i=r-k}^r \lambda_{[i]}(\mathbf{x})$ .
2.  $\max_{[k]}(\lambda(\mathbf{x})) = \sum_{i=1}^k \lambda_{[i]}(\mathbf{x})$ .
3.  $\alpha_1 \lambda_{[1]}(\lambda(\mathbf{x})) + \cdots + \alpha_k \lambda_{[k]}(\lambda(\mathbf{x}))$ , and  $\alpha_1 \lambda_{[1]}(\lambda(\mathbf{x})) + \cdots + \alpha_k \lambda_{[k]}(\lambda(\mathbf{x}))$ , where  $\alpha_1 \geq \cdots \geq \alpha_k > 0$ .

4. The elementary symmetric functions  $\sigma_k(\lambda(\mathbf{x})) = \sum \lambda_{[i_1]}(\mathbf{x}) \cdots \lambda_{[i_k]}(\mathbf{x})$  where the sum is over all subsets of size  $k$  of  $\{1, \dots, r\}$ .
5. Power sums  $\sum_i \lambda_{[i]}^l(\mathbf{x})$ .

Note that the elementary symmetric function  $(-1)^k \sigma_k(\lambda(\mathbf{x}))$  is the coefficient  $a_k(\mathbf{x})$  of the characteristic polynomial of  $\mathbf{x}$ , in particular,  $\det(\mathbf{x})$  is a spectral function, and thus so is  $\ln \det(\mathbf{x})$ . We will use  $-\ln \det(\mathbf{x})$  as a barrier function for the symmetric cone associated with the Jordan algebra  $(\mathbb{E}, \circ)$ .

Michel Baes has done a comprehensive study of spectral functions of elements of formally real Jordan algebras, [11]. It can be shown that minimization of convex, spectral functions can be expressed as optimization over symmetric cones. This fact is known for symmetric functions over real symmetric, or hermitian matrices, see for example [12]. However, the arguments can easily be generalized to formally real Jordan algebras. For example maximization of  $\det(\mathbf{x})$  can be expressed as an optimization over symmetric cones.

## 11.5 Applications in Optimization: Non-degeneracy and Interior Point Methods

In this section we apply the machinery developed in the previous sections to optimization problems (11.1) over symmetric cones. *Symmetric conic programming (scp)*<sup>6</sup> is the conic optimization problem where the underlying cone is symmetric, and therefore is the cone of squares of some formally real Jordan algebra. Since symmetric cones are self dual, the *standard form* **scp** problems can be written as:

Primal	Dual	
$\min \quad \langle \mathbf{c}_1, \mathbf{x}_1 \rangle + \cdots + \langle \mathbf{c}_n, \mathbf{x}_n \rangle$	$\max \quad \mathbf{b}^\top \mathbf{y}$	
s.t. $A_1 \mathbf{x}_1 + \cdots + A_n \mathbf{x}_n = \mathbf{b}$	s.t. $A_i^\top \mathbf{y} + \mathbf{s}_i = \mathbf{c}_i \quad i = 1, \dots, n$	(11.24)
$\mathbf{x}_i \succcurlyeq 0$	$\mathbf{s}_i \succcurlyeq 0$	

where each “ $\succcurlyeq$ ” is the partial order induced by a *simple* symmetric cone  $\mathcal{K}_{\mathbb{E}_i}$ , that is one whose underlying formally real Jordan algebra  $(\mathbb{E}_i, \circ)$  is simple. We also write  $\mathbb{E} = \mathbb{E}_1 \oplus \cdots \oplus \mathbb{E}_n$ , for the direct sum Jordan algebra, and  $A = [A_1, \dots, A_n]$ ,  $\mathbf{c} = [\mathbf{c}_1; \dots; \mathbf{c}_n]$ . Finally,  $\langle \cdot, \cdot \rangle$  is some inner product on  $\mathbb{E}_i$ . It can be shown that in simple Jordan algebras all inner products are related by a constant factor. Therefore, without loss of generality we may assume  $\langle \mathbf{x}, \mathbf{y} \rangle = \text{trace}(L(\mathbf{x} \circ \mathbf{y}))$ .

The following theorem is the *strong duality theorem* which is specialized from general conic programming (see for example, [12]).

---

<sup>6</sup>We write SDP as an abbreviation of semidefinite programming over matrices, and **scp** for symmetric conic programming.

**Theorem 11.13.** *Let in the symmetric conic programming problem (11.24) both Primal and Dual problems be feasible. Furthermore, assume that there is a feasible  $\mathbf{x}_i > 0$  for Primal and a feasible  $(\mathbf{y}_i, \mathbf{s}_i)$  for Dual, with  $\mathbf{s}_i > 0$ . Then if  $\mathbf{x}^* = (\mathbf{x}_1^*; \dots; \mathbf{x}_n^*)$  is optimal for Primal and  $(\mathbf{y}^*, \mathbf{s}^*)$ , with  $\mathbf{s}^* = (\mathbf{s}_1^*; \dots; \mathbf{s}_n^*)$ , is optimal for Dual, then we have  $\mathbf{x}_i^* \circ \mathbf{s}_i^* = \mathbf{0}$  for  $i = 1, \dots, n$ .*

In fact, the complementary set of a symmetric cone may be characterized as follows.

**Lemma 11.16.** *The complementary set associated to a symmetric cone, with the underlying formally real Jordan algebra  $(\mathbb{E}, \circ)$  is given by*

$$C(\mathcal{K}_{\mathbb{E}}) = \{(\mathbf{x}, \mathbf{s}) \mid \mathbf{x}, \mathbf{s} \in \mathcal{K}_{\mathbb{E}}, \mathbf{x} \circ \mathbf{s} = \mathbf{0}\}.$$

The proof is straightforward and can be found for example in [9].

To keep our discussion simple, we assume that both Primal and Dual problems are feasible and both contain strictly positive definite feasible points.

In the remainder of this section, we first briefly review the notions of primal and dual nondegeneracy and strict complementarity in **scp**, and then review the interior point approach to solving **scp** problems.

### 11.5.1 Nondegeneracy and Strict Complementarity in **scp**

The notion of nondegeneracy and strict complementarity for semidefinite programming and concrete conditions to check them were developed in [3] and later in Pataki [45]. For second order conic programming these results were given in the paper of the author and Goldfarb [1]. Faybusovich generalized these notions to symmetric conic programming in [14].

We abbreviate the Primal and Dual problems (11.24) by combining the simple cones of  $\mathbb{E}_i$ , matrices  $A_i$ , and  $\mathbf{c}_i$  as follows:

$$\begin{aligned} \text{Primal: } & \min \{ \langle \mathbf{c}, \mathbf{x} \rangle \mid \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0} \} \\ \text{Dual: } & \max \{ \mathbf{b}^\top \mathbf{y} \mid \mathbf{A}^\top \mathbf{y} + \mathbf{s} = \mathbf{c}, \mathbf{s} \geq \mathbf{0} \} \end{aligned} \quad (11.25)$$

First, we start with strict complementarity. Recall that in linear programming when  $\mathbf{x}^*$  is optimal for Primal, and  $\mathbf{s}^*$  is the optimal slack for Dual, then complementary slackness implies that  $x_i s_i = 0$ , for  $i = 1, \dots, n$ . We say strict complementarity holds when for each pair  $(x_i, s_i)$  exactly one equals zero. In linear programming it can be shown that whenever both Primal and Dual are feasible then there is always a pair of optimal solutions  $\mathbf{x}^*, \mathbf{s}^*$  satisfying strict complementarity.

This notion is extended to **scp** as follows. Let  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{s}^*)$  be optimal for Primal and Dual. Then as stated earlier, since  $\mathbf{x}^* \circ \mathbf{s}^* = \mathbf{0}$ ,  $\mathbf{x}$  and  $\mathbf{s}$  operator commute and

must share a common Jordan frame. Let <sup>7</sup>  $\mathbf{x}^* = \lambda_1 \mathbf{c}_1 + \dots + \lambda_r \mathbf{c}_r$  and  $\mathbf{s}^* = \omega_1 \mathbf{c}_1 + \dots + \omega_r \mathbf{c}_r$ . Then complementary slackness theorem implies  $\lambda_i \omega_i = 0$ . Then the natural extension of strict complementarity is to have exactly one of  $\lambda_i$  and  $\omega_i$  be zero. It is immediate that this characterization is equivalent to the following definition.

**Definition 11.15.** Let  $\mathbf{x}^*$  and  $\mathbf{s}^*$  be the optimal solutions of Primal and Dual problems over the symmetric cone  $\mathcal{K}_{\mathbb{E}}$ , where  $\text{rk}(\mathbb{E}) = r$ . Then we say  $\mathbf{x}^*$  and  $\mathbf{s}^*$  satisfy *strict complementarity* if  $\text{rk}(\mathbf{x}^*) + \text{rk}(\mathbf{s}^*) = r$ .

In particular, strict complementarity in (11.24) implies that, if  $\mathbf{x}_i^* > 0$  then  $\mathbf{s}_i^* = \mathbf{0}$ , and if  $\mathbf{s}_i^* > 0$  then  $\mathbf{x}_i^* = \mathbf{0}$ . On the other hand if both  $\mathbf{x}_i^*$  and  $\mathbf{s}_i^*$  are singular, then  $\text{rk}(\mathbf{x}_i^*) + \text{rk}(\mathbf{s}_i^*) = \text{rk}(\mathbb{E}_i)$ .

Unlike in linear programming one can find **sep**'s where there are no optimal solutions satisfying strict complementarity. It has been the author's experience that problems for which strict complementarity does not hold at the optimal solution present the greatest challenge in terms of numerical stability. Most software packages for solving SDP and SOCP that the author has worked with break down after finding the solution up to three or four digits of accuracy, even for fairly small size problems.

We now discuss the notion of primal and dual nondegeneracy. Note that the feasible region of Primal is the intersection of the affine space  $\mathcal{A} = \{\mathbf{x} \mid A\mathbf{x} = \mathbf{b}\}$  and the cone  $\mathcal{K}_{\mathbb{E}}$ . We say that a point  $\mathbf{x} \in \mathcal{A} \cap \mathcal{K}_{\mathbb{E}}$  is *primal nondegenerate* if  $\mathcal{A}$  and  $\mathcal{K}_{\mathbb{E}}$  intersect transversally at  $\mathbf{x}$ . Let the tangent space at  $\mathbf{x}$  to  $\mathcal{K}_{\mathbb{E}}$  be  $\mathcal{T}_{\mathbf{x}}$ . Since  $\mathcal{A}$  is an affine space, the tangent space at any point on it is the linear space parallel to it; this linear space is  $\text{Ker } A$ .

**Definition 11.16.** A primal feasible point  $\mathbf{x}$  is primal nondegenerate if  $\mathcal{T}_{\mathbf{x}} + \text{Ker } A = \mathbb{E}$  or equivalently  $\mathcal{T}_{\mathbf{x}}^\perp \cap \text{Range } A = \{\mathbf{0}\}$ .

Likewise, nondegeneracy at a feasible point  $(\mathbf{y}, \mathbf{s})$  for Dual can be defined.

**Definition 11.17.** Let  $\mathbf{y}, \mathbf{s}$  be feasible for Dual. Then this point is *dual nondegenerate* if  $\mathcal{T}_{\mathbf{s}} + \text{Range } A^\top = \mathbb{E}$  or equivalently,  $\mathcal{T}_{\mathbf{s}}^\perp \cap \text{Ker } A^\top = \{\mathbf{0}\}$ .

These definitions actually make sense for general conic optimization problems, but in the case of symmetric cones, the tangent spaces  $\mathcal{T}_{\mathbf{x}}$  and  $\mathcal{T}_{\mathbf{s}}$  can be characterized using Peirce decomposition.

Let the point  $\mathbf{x} \in \mathcal{K}_{\mathbb{E}}$  where its version 2 spectral decomposition  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \dots + \lambda_k \mathbf{c}_k$ , with  $\lambda_i > 0$  for  $i = 1, \dots, k$ , therefore the remaining  $r - k$  eigenvalues are zero. If  $\mathbf{x} \in \text{Int } \mathcal{K}_{\mathbb{E}}$ , then the tangent space is the entire space  $\mathbb{E}$ . If on the other hand  $k < r$ , then  $\mathbf{x} \in \text{bd } \mathcal{K}_{\mathbb{E}}$ , the boundary of  $\mathcal{K}_{\mathbb{E}}$ .

**Theorem 11.14.** Let  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \dots + \lambda_k \mathbf{c}_k$ , with  $\lambda_i > 0$ , and each  $\mathbf{c}_i$  a primitive idempotent. Let  $\mathbf{c} = \mathbf{c}_1 + \dots + \mathbf{c}_k$ . Then the  $\mathcal{T}_{\mathbf{x}}^\perp = \mathbb{E}_0(\mathbf{c})$ , and thus  $\mathcal{T}_{\mathbf{x}} = \mathbb{E}_1(\mathbf{c}) \oplus \mathbb{E}_{\frac{1}{2}}(\mathbf{c})$ .

Therefore, we get the following characterization of nondegeneracy, see [14].

<sup>7</sup>In the remainder of this chapter the order of eigenvalues is not essential. Thus, to avoid writing  $\lambda_{\pi_i}$  and  $\omega_{\pi_i}$ , we will simply write  $\lambda_i$  and  $\omega_i$ .

**Corollary 11.1.** 1. Let  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \dots + \lambda_k \mathbf{c}_k$ , with  $\lambda_i > 0$ , and  $\mathbf{c}_i$  orthogonal primitive idempotents, and  $\mathbf{c} = \mathbf{c}_1 + \dots + \mathbf{c}_k$ . If  $\mathbf{x}$  is primal feasible then  $\mathbf{x}$  is primal nondegenerate if  $\mathbb{E}_0(\mathbf{c}) \cap \text{Range } A = \{\mathbf{0}\}$ .

2. Let  $\mathbf{s} = \omega_1 \mathbf{d}_1 + \dots + \omega_l \mathbf{d}_l$  with  $\omega_i > 0$ ,  $\mathbf{d}_i$  be a set of orthogonal primitive idempotents, and  $\mathbf{d} = \mathbf{d}_1 + \dots + \mathbf{d}_l$ . Then  $\mathbf{s}$  is dual nondegenerate if  $\mathbb{E}_0(\mathbf{d}) \cap \text{Ker } A^\top = \{\mathbf{0}\}$ .

It can be shown that primal nondegeneracy implies that Dual has a unique solution, and dual nondegeneracy implies that Primal has a unique solution. But, unlike in linear programming, it is possible to find **scp** problems which are both primal and dual nondegenerate at the optimum, and yet strict complementarity does not hold.

Let  $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{s}^*)$  be the optimum solution of an optimization problem over a symmetric cone  $\mathcal{K}_{\mathbb{E}}$ , and let  $\mathbf{x}^*$  and  $(\mathbf{y}^*, \mathbf{s}^*)$  be both nondegenerate and satisfy strict complementarity conditions. Then it can be shown that the system of equations (11.4) is nonsingular at the optimal point, see [14]. A consequence of this observation is that a number of algorithms depending on Newton's method—in particular interior point methods—are well-behaved. (See also [55] for the well-behaved nature of the Newton system for the  $Q$  method.)

### 11.5.2 Interior Point Methods

The function  $-\sum_i \ln \det(\mathbf{x}_i)$  is a barrier function for the optimization problem (11.24). Let  $\mathbf{x}_\mu = (x_{1\mu}, \dots, x_{n\mu})$  be  $\mu$ -optimal for Primal, and  $(\mathbf{y}_\mu, \mathbf{s}_\mu)$ , where  $\mathbf{s}_\mu = (s_{1\mu}, \dots, s_{n\mu})$ , be  $\mu$ -optimal for Dual in (11.24). Setting, as before,  $A = [A_1, \dots, A_n]$ , and recalling that  $\nabla_{\mathbf{x}} \ln \det \mathbf{x} = \mathbf{x}^{-1}$ , we arrive at the following relaxed system of equations which must be satisfied by  $\mu$ -optimal solutions:

$$\begin{aligned} A\mathbf{x} &= \mathbf{b} \\ A^\top \mathbf{y} + \mathbf{s} &= \mathbf{c} \\ \mathbf{x} \circ \mathbf{s} &= \mu \mathbf{e} \end{aligned} \tag{11.26}$$

Let us define  $\mathbf{r}_p = \mathbf{b} - A\mathbf{x}$ ,  $\mathbf{r}_d = \mathbf{c} - A^\top \mathbf{y} - \mathbf{s}$ , and  $\mathbf{r}_c = \mu \mathbf{e} - \mathbf{x} \circ \mathbf{s}$ . When  $\mathbf{x}$  is feasible for Primal and  $(\mathbf{y}, \mathbf{s})$  is feasible for Dual, then  $\mathbf{r}_p = \mathbf{0}$  and  $\mathbf{r}_d = \mathbf{0}$ . It can be shown that for  $\mu > 0$  the system (11.26) has a unique solution, which is the  $\mu$ -optimal point.

**Definition 11.18.** The space curve of points  $(\mathbf{x}_\mu, \mathbf{y}_\mu, \mathbf{s}_\mu)$  parameterized by  $\mu$  is called the (infeasible) *central path*. If  $\mathbf{r}_p = \mathbf{0}$  and  $\mathbf{r}_d = \mathbf{0}$ , then the space curve is called the feasible central path or simply the central path.

Note that because  $\mathbf{x} \circ \mathbf{s} = \mu \mathbf{e}$  the primal point  $\mathbf{x}$  and the dual slack point  $\mathbf{s}$  operator commute on the central path.

There are numerous interior point algorithms developed for semidefinite programming, and the vast majority of them can be extended to **scp** problems. In [9] a

pair of algorithms (developed originally by Ye for linear programming [56, 57] and later extended by the author to semidefinite programming [7, 8]) based on potential reduction methods was presented for **scp** problems. Since this method is completely developed in [9] we will not discuss it here any further.

Another class of problems are known as *path-following* methods, where by “path” we mean the central path of Definition 11.18 based on (11.26). As mentioned earlier applying the Newton method to the system (11.26), we get the following linear system of equations

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^\top & I \\ L(\mathbf{s}) & 0 & L(\mathbf{x}) \end{pmatrix} \begin{pmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\mathbf{s} \end{pmatrix} = \begin{pmatrix} \mathbf{b} - A\mathbf{x} \\ \mathbf{c} - A^\top\mathbf{y} - \mathbf{s} \\ \mu\mathbf{e} - \mathbf{x} \circ \mathbf{s} \end{pmatrix}. \quad (11.27)$$

(Recall that in Jordan algebras, because of the commutative property,  $R_\circ(\mathbf{x}) = L_\circ(\mathbf{x}) = L(\mathbf{x})$ .) The idea is that if  $(\mathbf{x}, \mathbf{y}, \mathbf{s})$  is our current estimate of the solution to the **scp** problem then after solving this system for  $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$  we obtain a new estimate  $(\mathbf{x} + \alpha\Delta\mathbf{x}, \mathbf{y} + \beta\Delta\mathbf{y}, \mathbf{s} + \gamma\Delta\mathbf{s})$ . The system (11.27) can be explicitly solved for the solution  $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$  as follows:

$$\Delta\mathbf{y} = \left( AL^{-1}(\mathbf{s})L(\mathbf{x})A \right)^{-1} \left( \mathbf{r}_p + AL^{-1}(\mathbf{s})(L(\mathbf{x})\mathbf{r}_d - \mathbf{r}_c) \right) \quad (11.28)$$

$$\Delta\mathbf{s} = \mathbf{r}_d - A^\top\Delta\mathbf{y} \quad (11.29)$$

$$\Delta\mathbf{x} = -L^{-1}(\mathbf{s}) \left( L(\mathbf{x})\Delta\mathbf{s} - \mathbf{r}_c \right) \quad (11.30)$$

Here too, our starting points are algorithms devised for linear programming. Though there are numerous variations, the primal-dual analysis described in Monteiro and Adler [32] is one of the simplest. However, we encounter difficulties in attempting to generalize this analysis to second order conic or semidefinite programming, and naturally also to symmetric conic programming. First let us see what happens in the special case of linear programming. In this case  $L(\mathbf{x}) = \text{Diag}(\mathbf{x})$  a diagonal matrix. However, we know that in this case *any*  $\mathbf{x}$  and  $\mathbf{s}$  operator commute, as product of diagonal matrices commute. This is due to the fact the Jordan algebra associated with the nonnegative orthant is the direct sum algebra  $\mathbb{R} \oplus \cdots \oplus \mathbb{R}$ , which is an associative and commutative algebra. This property is used in a crucial way in the analysis of Monteiro and Adler.

Going now to symmetric cones,  $L(\mathbf{x})$  and  $L(\mathbf{s})$  operator commute if  $(\mathbf{x}, \mathbf{y}, \mathbf{s})$  is on the central path; but otherwise they generally don’t. Therefore, extending the analysis of Monteiro and Adler (and any other primal-dual analysis developed for linear programming, for that matter) to symmetric cones turns out to be considerably more complicated. These extensions have nevertheless been done, and as a byproduct, many different types of primal-dual methods for general symmetric cones have been developed. We now outline some classes of such algorithms, and show how the techniques of formally real Jordan algebras adds a degree of elegance and transparency to the analysis.

We start with the notion of *neighborhoods* around the central path. Let us consider the formulation (11.24), with  $\mathbf{x} = (\mathbf{x}_1; \dots; \mathbf{x}_n)$ , and likewise,  $\mathbf{s} = (\mathbf{s}_1; \dots; \mathbf{s}_n)$ . Let  $\mathbf{w} = Q_{\mathbf{x}^{1/2}} \mathbf{s}$ . Consider the following three *centrality measures* defined for  $(\mathbf{x}, \mathbf{s}) \in \text{Int}\mathcal{K}_{\mathbb{E}} \times \text{Int}\mathcal{K}_{\mathbb{E}}$ :

$$d_F(\mathbf{x}, \mathbf{s}) \stackrel{\text{def}}{=} \|\mathbf{Q}_{\mathbf{x}^{1/2}} \mathbf{s} - \mu \mathbf{e}\|_F = \sqrt{\sum_{i=1}^r (\lambda_i(\mathbf{w}) - \mu)^2} \quad (11.31)$$

$$d_2(\mathbf{x}, \mathbf{s}) \stackrel{\text{def}}{=} \|\mathbf{Q}_{\mathbf{x}^{1/2}} \mathbf{s} - \mu \mathbf{e}\|_2 = \max_{i=1, \dots, r} |\lambda_i(\mathbf{w}) - \mu| \quad (11.32)$$

$$= \max\{\lambda_{[1]}(\mathbf{w}) - \mu, \mu - \lambda_{[r]}(\mathbf{w})\}$$

$$d_{-\infty}(\mathbf{x}, \mathbf{s}) \stackrel{\text{def}}{=} \mu - \lambda_{[r]}(\mathbf{w}) \quad (11.33)$$

Note that all of these centrality measures are in fact spectral functions of  $\mathbf{w}$ . Given a constant  $\gamma \in (0, 1)$ , we define the following neighborhoods of the central path

$$\mathcal{N}_F(\gamma) \stackrel{\text{def}}{=} \{(\mathbf{x}, \mathbf{s}, \mathbf{y}) \in \mathcal{F}^0(P) \times \mathcal{F}^0(D) \mid d_F(\mathbf{x}, \mathbf{s}) \leq \gamma\mu\} \quad (11.34)$$

$$\mathcal{N}_2(\gamma) \stackrel{\text{def}}{=} \{(\mathbf{x}, \mathbf{s}, \mathbf{y}) \in \mathcal{F}^0(P) \times \mathcal{F}^0(D) \mid d_2(\mathbf{x}, \mathbf{s}) \leq \gamma\mu\} \quad (11.35)$$

$$\mathcal{N}_{-\infty}(\gamma) \stackrel{\text{def}}{=} \{(\mathbf{x}, \mathbf{s}, \mathbf{y}) \in \mathcal{F}^0(P) \times \mathcal{F}^0(D) \mid d_{-\infty}(\mathbf{x}, \mathbf{s}) \leq \gamma\mu\}, \quad (11.36)$$

where the sets of interior feasible points are

$$\mathcal{F}^0(P) \stackrel{\text{def}}{=} \{\mathbf{x} \in \mathbb{E} \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \in \text{Int}\mathcal{K}_{\mathbb{E}}\},$$

$$\mathcal{F}^0(D) \stackrel{\text{def}}{=} \{(\mathbf{s}, \mathbf{y}) \in \mathbb{E} \times \mathbb{R}^m \mid A^T \mathbf{y} + \mathbf{s} = \mathbf{c}, \mathbf{s} \in \text{Int}\mathcal{K}_{\mathbb{E}}\}.$$

To justify these definitions, recall that the  $d_{\bullet}(\mathbf{x}, \mathbf{s})$  are functions of  $x_j s_j$  in LP and  $\lambda_i(XS)$ , i.e. eigenvalues of  $XS$  in SDP. However, note that  $\lambda_i(XS) = \lambda_i(X^{1/2} S X^{1/2})$ . As we know  $Q_{\mathbf{x}}$  is the counterpart of the operator that sends  $Y$  to  $XYX$  in matrix algebra. Therefore,  $Q_{\mathbf{x}^{1/2}} \mathbf{s}$  is the counterpart of  $X^{1/2} S X^{1/2}$ . Intuitively, these neighborhoods define a region around the central path which becomes narrower as  $\mu$  gets smaller. The goal of the algorithm designer is to find a sequence of points  $(\mathbf{x}_i, \mathbf{y}_i, \mathbf{s}_i)$  that stay in the relevant neighborhood as  $i$  increases, while, at the same time, the duality gap  $\langle \mathbf{x}_i, \mathbf{s}_i \rangle$  tends to zero. In case of infeasible methods, one would require that  $\mathbf{r}_p$  and  $\mathbf{r}_d$  also tend to zero.

Clearly,

$$d_F(\mathbf{x}, \mathbf{s}) \geq d_2(\mathbf{x}, \mathbf{s}) \geq d_{-\infty}(\mathbf{x}, \mathbf{s}),$$

which immediately implies

$$\mathcal{N}_F(\gamma) \subseteq \mathcal{N}_2(\gamma) \subseteq \mathcal{N}_{-\infty}(\gamma) \subseteq \mathcal{K} \times \mathcal{K}.$$

Therefore, intuitively it may be more desirable to work with  $\mathcal{N}_{-\infty}$  since it is a larger neighborhood, and we have more room to move in the Newton direction at each iteration. However, it turns out that proving polynomial time convergence for this neighborhood is more difficult. In fact the best theoretical results are known for the smallest neighborhood  $\mathcal{N}_F$ .

We can apply the Newton method as described in (11.27); this approach is known as the AHO method in literature since it was first proposed for semidefinite programming in [4]. An analysis devised by Monteiro [35] for semidefinite programming shows how to generate a sequence of points that remain in  $\mathcal{N}_F$  neighborhood and in  $O(\sqrt{n})$  iterations reduce the size of duality gap by a constant factor. This analysis was extended to second order conic programming by Monteiro and Tsuchiya [36]. Finally Schmieta and Alizadeh extended this analysis to those symmetric conic optimization problems whose underlying Jordan algebras are special in [47].

To achieve an easier analysis, for example by ensuring that the Schur complement  $A(L^{-1}(\mathbf{s})L(\mathbf{x}))A^\top$  is a symmetric positive semidefinite matrix, we may use carefully chosen automorphisms of  $\mathcal{K}_{\mathbb{E}}$  to *scale* the problem. The idea is to first apply this automorphism to the problem, and then apply one iteration of the Newton's method as described in (11.27) in the scaled space. Once the solution of the this system yields a new estimate in the transformed space, we then apply the inverse automorphism to get a new estimate in the original space.

To see how scaling works recall that in general for every invertible  $\mathbf{p} \in \mathbb{E}$  the quadratic representation  $Q_{\mathbf{p}}$  is an automorphism of  $\mathcal{K}_{\mathbb{E}}$ . Let  $\mathbf{p} > \mathbf{0}$ . With respect to  $\mathbf{p}$ , define

$$\widetilde{\mathbf{u}} \stackrel{\text{def}}{=} Q_{\mathbf{p}}\mathbf{u} \text{ and } \underline{\mathbf{u}} \stackrel{\text{def}}{=} Q_{\mathbf{p}^{-1}}\mathbf{u}.$$

Note that since  $Q_{\mathbf{p}}Q_{\mathbf{p}^{-1}} = I$ , the operators  $\widetilde{\cdot}$  and  $\underline{\cdot}$  are inverses of each other. With the change of variables:  $\mathbf{x} \rightarrow \widetilde{\mathbf{x}}$ , both  $\mathcal{K}_{\mathbb{E}}$  and the duality gap remain invariant.

**Lemma 11.17.** *Let  $\mathbf{x}$ ,  $\mathbf{s}$ , and  $\mathbf{p}$  be in some formally real Jordan algebra  $\mathbb{E}$ ,  $\mathbf{x}, \mathbf{s} > \mathbf{0}$ , and let  $\mathbf{p}$  be invertible. Then  $\mathbf{x} \circ \mathbf{s} = \alpha \mathbf{e}$  if and only if  $(Q_{\mathbf{p}}\mathbf{x}) \circ (Q_{\mathbf{p}^{-1}}\mathbf{s}) = \alpha \mathbf{e}$ .*

Proofs of this and other results in this section can be found in [48]. Therefore, we may write the relaxed complementarity condition  $\mathbf{x} \circ \mathbf{s} = \mu \mathbf{e}$  as  $(Q_{\mathbf{p}}\mathbf{x}) \circ (Q_{\mathbf{p}^{-1}}\mathbf{s}) = \mu \mathbf{e}$ . Thus the Newton system (11.27) can be equivalently written as

$$\begin{aligned} \underline{A}\widetilde{\Delta\mathbf{x}} &= \mathbf{b} - \underline{A}\widetilde{\mathbf{x}} \\ \underline{A}^T\Delta\mathbf{y} + \underline{\Delta\mathbf{s}} &= \underline{\mathbf{c}} - \underline{\mathbf{s}} - \underline{A}^T\mathbf{y} \\ \widetilde{\Delta\mathbf{x}} \circ \underline{\mathbf{s}} + \widetilde{\mathbf{x}} \circ \underline{\Delta\mathbf{s}} &= \mu \mathbf{e} - \widetilde{\mathbf{x}} \circ \underline{\mathbf{s}}, \end{aligned} \tag{11.37}$$

where  $\underline{A} = AQ_{\mathbf{p}^{-1}}$ . This scaling has the following additional properties.

**Lemma 11.18.** *With respect to a  $\mathbf{p} > \mathbf{0}$  we have:*

- i.  $\langle \mathbf{x}, \mathbf{s} \rangle = \langle \widetilde{\mathbf{x}}, \underline{\mathbf{s}} \rangle$ .
- ii. For each vector  $\mathbf{u}$ ,  $A\mathbf{u} = \mathbf{0}$  if and only if  $\underline{A}\widetilde{\mathbf{u}} = \mathbf{0}$ .

- iii.  $d_{\bullet}(\mathbf{x}, \mathbf{s}) = d_{\bullet}(\tilde{\mathbf{x}}, \underline{\mathbf{s}})$  for the  $F$ , 2, and  $-\infty$  centrality measures.
- iv. Under the given scaling, the central path and the neighborhoods  $\mathcal{N}_F$ ,  $\mathcal{N}_2$  and  $\mathcal{N}_{-\infty}$  remain invariant.

**Lemma 11.19.**  $(\tilde{\Delta}\mathbf{x}, \Delta\mathbf{y}, \underline{\Delta}\mathbf{s})$  solves the system of equations (11.37) if and only if  $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$  solves

$$\begin{aligned} A\Delta\mathbf{x} &= \mathbf{b} - A\mathbf{x} \\ A^T\Delta\mathbf{y} + \Delta\mathbf{s} &= \mathbf{c} - A^T\mathbf{y} - \mathbf{s} \\ (Q_p\Delta\mathbf{x}) \circ (Q_{p^{-1}}\mathbf{s}) + (Q_p\mathbf{x}) \circ (Q_{p^{-1}}\Delta\mathbf{s}) &= 2\mu\mathbf{e} - (Q_p\mathbf{x}) \circ (Q_{p^{-1}}\mathbf{s}). \end{aligned} \quad (11.38)$$

Note that  $(\tilde{\Delta}\mathbf{x}, \Delta\mathbf{y}, \underline{\Delta}\mathbf{s})$  is the result of applying Newton's method to the primal and dual feasibility and complementarity ( $\tilde{\mathbf{x}} \circ \underline{\mathbf{s}} = \mu\mathbf{e}$ ) relations arising from the scaled problem (11.26). The corresponding  $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$  is different from the one derived in (11.28), (11.29) and (11.30). Indeed the former depends on  $\mathbf{p}$  and the latter results as a special case when  $\mathbf{p} = \mathbf{e}$ .

We see that for each choice of  $\mathbf{p}$  we get a different set of directions. This class of directions is known as the *Monteiro–Zhang family of directions*, as they were originally developed for semidefinite programming in [34] and [58]. The AHO method is the special case where  $\mathbf{p} = \mathbf{e}$ . The  $O(\sqrt{r})$  convergence rate analysis given in [35] and [47] actually is valid for the entire Monteiro–Zhang class, but as mentioned, the neighborhood used to prove this rate of convergence is  $\mathcal{N}_F$ , the smallest of the group.

We saw when  $L(\mathbf{x})$  and  $L(\mathbf{s})$  do not commute the analysis becomes considerably more complicated. A workaround is to find a scaling  $\mathbf{Q}_p$  where in the transformed space  $\tilde{\mathbf{x}}$  and  $\underline{\mathbf{s}}$  operator commute. Define,

$$\mathfrak{C}(\mathbf{x}, \mathbf{s}) = \left\{ \mathbf{p} \mid \mathbf{p} \text{ nonsingular, } \mathbf{Q}_p\mathbf{x} \text{ and } \mathbf{Q}_{p^{-1}}\mathbf{s} \text{ operator commute} \right\}.$$

This is a subclass of the Monteiro–Zhang family of search directions called the *commutative class*.

It is clear that when  $\mathbf{p} = \mathbf{e}$  the resulting direction is not in general in the commutative class. However, the following choices of  $\mathbf{p}$  result in commutative directions:

$$\begin{aligned} \mathbf{p} &= \mathbf{s}^{1/2}, \quad \mathbf{p} = \mathbf{x}^{-1/2}, \\ \mathbf{p} &= \left[ Q_{\mathbf{x}^{1/2}}(Q_{\mathbf{x}^{1/2}}\mathbf{s})^{-1/2} \right]^{-1/2} = \left[ Q_{\mathbf{s}^{-1/2}}(Q_{\mathbf{s}^{1/2}}\mathbf{x})^{1/2} \right]^{-1/2}. \end{aligned}$$

The first two are generalizations of directions first given in [24], [30], and [34] for SDP. The third one is the Nesterov and Todd (NT) direction given in [43, 44]. For  $\mathbf{p} = \mathbf{s}^{1/2}$  we have:

$$\underline{\mathbf{s}} = Q_{\mathbf{p}^{-1}}\mathbf{s} = Q_{\mathbf{s}^{-1/2}}(\mathbf{s}^{1/2})^2 = (Q_{\mathbf{s}^{-1/2}}Q_{\mathbf{s}^{1/2}})\mathbf{e} = \mathbf{e}.$$

Similarly for  $\mathbf{p} = \mathbf{x}^{-1/2}$  it can be shown that  $\tilde{\mathbf{x}} = \mathbf{e}$ . For the Nesterov–Todd direction  $\mathbf{p}$ ,

$$\begin{aligned} Q_{\mathbf{p}}^2 \mathbf{x} &= Q_{\mathbf{p}^2} \mathbf{x} = \left( Q_{Q_{\mathbf{x}^{1/2}}(Q_{\mathbf{x}^{1/2}} \mathbf{s})^{-1/2}}^{-1} \right) \mathbf{x} = \left( Q_{Q_{\mathbf{x}^{-1/2}}(Q_{\mathbf{x}^{1/2}} \mathbf{s})^{1/2}} \right) \mathbf{x} \\ &= (Q_{\mathbf{x}^{-1/2}} Q_{(Q_{\mathbf{x}^{1/2}} \mathbf{s})^{1/2}} Q_{\mathbf{x}^{-1/2}}) \mathbf{x} = (Q_{\mathbf{x}^{-1/2}} Q_{(Q_{\mathbf{x}^{1/2}} \mathbf{s})^{1/2}}) \mathbf{e} = (Q_{\mathbf{x}^{-1/2}} Q_{\mathbf{x}^{1/2}}) \mathbf{s} = \mathbf{s}; \end{aligned}$$

hence  $\tilde{\mathbf{x}} = Q_{\mathbf{p}} \mathbf{x} = Q_{\mathbf{p}^{-1}} \mathbf{s} = \underline{\mathbf{s}}$ . It follows that in each of these cases  $\tilde{\mathbf{x}}$  and  $\underline{\mathbf{s}}$  operator commute.

We now briefly sketch some known results. Suppose that  $(\mathbf{x}, \mathbf{y}, \mathbf{s})$  is a feasible point in the appropriate neighborhood and that  $(\Delta \mathbf{x}, \Delta \mathbf{y}, \Delta \mathbf{s})$  is the direction obtained by solving the system (11.38) with  $\mu = \frac{\langle \mathbf{x}, \mathbf{s} \rangle}{n} \sigma$  for some constant  $\sigma \in (0, 1)$ . This direction, in turn, leads to the new feasible point  $(\mathbf{x} + \Delta \mathbf{x}, \mathbf{y} + \Delta \mathbf{y}, \mathbf{s} + \Delta \mathbf{s})$ . Then

$$\begin{aligned} \langle (\mathbf{x} + \Delta \mathbf{x}), (\mathbf{s} + \Delta \mathbf{s}) \rangle &= \langle \mathbf{x}, \mathbf{s} \rangle + \langle \Delta \mathbf{x}, \mathbf{s} \rangle + \langle \mathbf{x}, \Delta \mathbf{s} \rangle + \langle \Delta \mathbf{x}, \Delta \mathbf{s} \rangle \\ &= \langle \mathbf{x}, \mathbf{s} \rangle + \sigma \frac{\langle \mathbf{x}, \mathbf{s} \rangle}{n} \text{Trace } L(\mathbf{e}) - \langle \mathbf{x}, \mathbf{s} \rangle + \langle \Delta \mathbf{x}, \Delta \mathbf{s} \rangle \\ &= \sigma \langle \mathbf{x}, \mathbf{s} \rangle + \langle \Delta \mathbf{x} \Delta \mathbf{s} \rangle \end{aligned}$$

In feasible point approaches  $\mathbf{r}_p = \mathbf{0}$  and  $\mathbf{r}_d = \mathbf{0}$ , which implies  $\langle \Delta \mathbf{x}, \Delta \mathbf{s} \rangle = 0$ . Therefore, at each iteration the duality gap is reduced by a factor of  $\sigma$ . The harder part is to choose  $\sigma$  and  $\gamma$  in such a way that the new point remains in the corresponding neighborhood  $\mathcal{N}_*(\gamma)$ , thus ensuring feasibility throughout all iterations.

1. For the neighborhood  $\mathcal{N}_F(\gamma)$  it can be shown that for all  $\mathbf{p}$  the new point is also in  $\mathcal{N}_F(\gamma)$  if we choose  $\sigma = \left(1 - \frac{\delta}{\sqrt{\gamma}}\right)$ , for some  $\gamma, \delta \in (0, 1)$ . In this case, as stated earlier, the iteration complexity of the algorithm is  $\mathcal{O}(\sqrt{r})$ , see [35] for SDP, [36] for SOCP, and [47, 48, 50] for optimization over representable symmetric cones.
2. For the commutative directions in  $\mathcal{N}_2$  and  $\mathcal{N}_{-\infty}$ , one can show that the iteration complexity is  $\mathcal{O}(kr)$ , again for  $\sigma = \left(1 - \frac{\delta}{\sqrt{\gamma}}\right)$  and suitable  $\gamma, \delta \in (0, 1)$ . Here,  $\kappa$  is the least upper bound on the condition number of the matrix  $G^{1/2}$  in  $\mathcal{N}_*(\gamma)$ , where  $G = L^{-1}(\underline{\mathbf{s}}) L(\tilde{\mathbf{x}})$ . For the Nesterov–Todd direction  $G = I$  the identity matrix, and so the iteration complexity is  $\mathcal{O}(r)$  in the  $\mathcal{N}_2$  and the  $\mathcal{N}_{-\infty}$  neighborhoods, respectively. When  $\mathbf{p} = \mathbf{x}^{-1/2}$  or  $\mathbf{p} = \mathbf{s}^{1/2}$  one can show that  $\kappa = \mathcal{O}(1)$  for  $\mathcal{N}_2(\gamma)$  and  $\kappa = \mathcal{O}(r)$  for  $\mathcal{N}_{-\infty}(\gamma)$ . Therefore, the iteration complexity is  $\mathcal{O}(r)$  and  $\mathcal{O}(r^{1.5})$ , respectively for  $\mathcal{N}_2(\gamma)$  and  $\mathcal{N}_{-\infty}(\gamma)$  ([37], [52], [47]). The complexity of non-commutative directions remains unresolved with respect to the  $\mathcal{N}_2(\gamma)$  and the  $\mathcal{N}_{-\infty}(\gamma)$  neighborhoods.

Finally, we should mention the so called  $Q$  method originally devised by Alizadeh et al. [2] for semidefinite programming. This algorithm was extended by Xia and Alizadeh to second order conic programming [54] and to symmetric conic programming [55]. The main idea is that instead of directly looking for  $\mathbf{x}$  and  $\mathbf{s}$ , we take advantage of the fact that at the optimum, since  $\mathbf{x} \circ \mathbf{s} = \mathbf{0}$ ,  $\mathbf{x}$  and  $\mathbf{s}$  operator

commute; thus they share a common Jordan frame. If we write  $\mathbf{x} = \lambda_1 \mathbf{c}_1 + \cdots + \lambda_r \mathbf{c}_r$  and  $\mathbf{s} = \omega_q \mathbf{c}_1 + \cdots + \omega_r \mathbf{c}_r$ . The algorithm then separately searches for corrections to the current  $\lambda_i$  and  $\omega_i$  and  $\mathbf{c}_i$ . The correction to eigenvalues is done through standard Newton method. But for Jordan frames, the correction is an automorphism in the  $K_{\mathbb{E}}$  subgroup. As it turns out, this subgroup is the Lie group associated to the Lie algebra of *derivations*<sup>8</sup> of  $\mathbb{E}$ . Since derivations form a linear space, the Newton method is applied to them. It can be shown [16] that for each derivation  $D$ ,  $\exp(D)$  is in  $\text{Aut}(\mathbb{E})$ . Here again Peirce decomposition, quadratic representations, spectral decomposition and exponential map come together in the development of the  $Q$  method; see [55] for details.

## 11.6 Conclusion

Formally real Jordan algebras are used in many other contexts useful to optimization and semidefinite programming. Gowda and Sznajder [17] generalize the concept of Schur complement. Cholesky factorization, and polar decomposition are extended in [16]. Also, Gowda et al. [18] have studied complementarity problems on symmetric cones. In addition to interior point methods there are other algorithms for SDP, SOCP, and sometimes for general convex problems. Many of these algorithms may be extended to **scp**. A representative work is presented in Baes's thesis [10] which extends Nesterov's smoothing algorithm, [39], from SDP to **scp**. Another example is extension of well-known SDP-representability of nonnegative (univariate) polynomials [29] and squared functional systems [38] to symmetric cones by Papp et al. [46]. For instance, optimization over the set of  $\mathbb{E}$ -valued semidefinite polynomials

$$\{\mathbf{p}(t) = \mathbf{a}_0 + \mathbf{a}_1 t + \cdots + \mathbf{a}_m t^m \mid \mathbf{p}(t) \succeq 0 \text{ for all } t \in \Delta\}$$

can be formulated as a  $\mathcal{K}_{\mathbb{E}}$ -optimization problem whenever  $\Delta$  is a finite union of connected subsets of  $\mathbb{R}$ .

It has been argued that inclusion of Jordan algebraic techniques does not contribute anything new, and that all results based on them had already been derived by other methods without their use. However, it is hard to argue against the power of Jordan algebraic tools in simplifying and unifying presentation of properties of certain conic optimization problems. Such techniques also unify in a transparent way design and analysis of algorithms for SDP, SOCP and related problems. One can only look at the work of Nesterov and Todd [43, 44] and compare their presentation to those based on Jordan algebraic methods, to appreciate transparency and simplicity afforded by these techniques.

---

<sup>8</sup>A linear transformation  $D$  on  $\mathbb{E}$  is a derivation if  $D(\mathbf{x} \circ \mathbf{y}) = (D(\mathbf{x})) \circ \mathbf{y} + (\mathbf{x}) \circ (D(\mathbf{y}))$  for all  $\mathbf{x}, \mathbf{y} \in \mathbb{E}$ .

We close this chapter by stating an open problem regarding complementarity set  $C(\mathcal{K}, \mathcal{K}^*)$ . The question is for what cones the set  $C(\mathcal{K}, \mathcal{K}^*)$  can be described entirely by bilinear forms. To our knowledge only symmetric cones and cones linearly isomorphic to them are known to have this property. Rudolf et al. [41] show that for some well-known classes of cones bilinear relations alone are not sufficient to characterize  $C(\mathcal{K}, \mathcal{K}^*)$ . Are there any non-symmetric cones which can be characterized solely by bilinear forms?

**Acknowledgements** The author greatly appreciates the thorough reading of this chapter by an anonymous reviewer, and correcting numerous stylistic and some mathematical errors present in earlier versions. This work was supported in part by the US National Science Foundation Grant number CMMI-0935305.

## References

1. Alizadeh, F., Goldfarb, D.: Second-Order Cone Programming. *Math. Programming Series B* **95**, 3–51 (2003)
2. Alizadeh, F., Haeberly, J.P., Overton, M.L.: A new primal-dual interior point method for semidefinite programming. *Proc. 5th SIAM Conference on Appl. Linear Algebra*, Snowbird, Utah (1994)
3. Alizadeh, F., Haeberly, J.P., Overton, M.L.: Complementarity and nondegeneracy in semidefinite programming. *Math. Programming* **77**(2) (1997)
4. Alizadeh, F., Haeberly, J.P., Overton, M.L.: Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results. *SIAM J. Optim.* **8**(3), 746–768 (1998)
5. Albert, A.: A Note on the Exceptional Jordan Algebra. *Proceedings of the National Academy of Sciences of the United States of America* **36**, 372–374 (1950)
6. Alizadeh, F.: Combinatorial optimization with interior point methods and semi-definite matrices. Ph.D. thesis, Computer Science Department, University of Minnesota, Minneapolis, Minnesota (1991)
7. Alizadeh, F.: Optimization Over Positive Semi-Definite Cone; Interior-Point Methods and Combinatorial Applications. In: Pardalos, P. (ed.) *Advances in Optimization and Parallel Computing*, North-Holland, Amsterdam (1992)
8. Alizadeh, F.: Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optim.* **5**(1), 13–51 (1995)
9. Alizadeh, F., Schmieta, S.H.: Symmetric Cones, Potential Reduction Methods and Word-By-Word Extensions. In: Saigal, R., Vandenberghe, L., Wolkowicz, H. (eds.) *Handbook of Semidefinite Programming, Theory, Algorithms and Applications*, pp. 195–233. Kluwer Academic Publishers (2000)
10. Baes, M.: Spectral Functions and Smoothing Techniques on Jordan Algebras. Ph.D. thesis, Université Catholique de Louvain (2006)
11. Baes, M.: Convexity and Differentiability Properties of Spectral Functions and Spectral Mappings on Euclidean Jordan Algebras. *Linear Algebra Appl.* **422**(2-3), 664–700 (2007)
12. Ben-Tal, A., Nemirovskiae, A.S.: *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2001)
13. Faybusovich, L.: Euclidean Jordan algebras and interior-point algorithms. *Positivity* **1**(4), 331–357 (1997)

14. Faybusovich, L.: Linear systems in Jordan algebras and primal-dual interior point algorithms. *Journal of Computational and Applied Mathematics* **86**, 149–175 (1997)
15. Faybusovich, L.: A Jordan-algebraic approach to potential-reduction algorithms. *Mathematische Zeitschrift* **239**, 117–129 (2002)
16. Faraut, J., Korányi, A.: Analysis on symmetric cones. Oxford University Press, Oxford, UK (1994)
17. Gowda, M.S., Sznajder, R.S.: Schur complements, Schur determinental and Haynsworth inertia formulas in Euclidean Jordan algebras. *Linear Algebra Appl.* **432** (2010)
18. Gowda, M.S., Sznajder, R.S., Tao, L.: Some  $\mathbf{P}$ -properties for linear transformations on Euclidean Jordan algebras. *Linear Algebra Appl.* **393**, 203–232 (2004)
19. Güler, O.: Barrier Functions in Interior Point Methods. *Math. of Oper. Res.* **21**, 860–885 (1996)
20. Güler, O.: Personal communication (1997)
21. Horn, R., Johnson, C.: Matrix analysis. Cambridge University Press, Cambridge (1985)
22. Horn, R., Johnson, C.: Topics in matrix analysis. Cambridge University Press, Cambridge (1990)
23. Hardy, G., Littlewood, J.E., Pólya, G.: Inequalities, 2nd ed., Cambridge University Press (1952)
24. Helmberg, C., Rendl, F., Vanderbei, R.J., Wolkowicz, H.: An interior-point method for semidefinite programming. *SIAM J. Optim.* **6**, 342–361 (1996)
25. Jacobson, N.: Structure and Representation of Jordan Algebras. Colloquium Publications, vol. XXXIX, American Mathematical Society, Providence, Rhode Island (1968)
26. Jacobson, N.: Basic Algebra I. Dover (2009)
27. Jordan, P., von Neumann, J., Wigner, E.: On algebraic generalization of the quantum mechanical formalism. *Ann. of Math.* **36**, 29–64 (1934)
28. Koecher, M.: The Minnesota Notes on Jordan Algebras and Their Applications. Springer-Verlag (1999). Edited by Kreig, A. and Walcher, S. based on Lectures given in The University of Minnesota (1960)
29. Karlin, S., Studden, W.: Tchebychev Systems: With Applications in Analysis and Statistics. Wiley interscience Publishers (1966)
30. Kojima, M., Shindoh, S., Hara, S.: Interior-point methods for the monotone linear complementarity problem in symmetric matrices. *SIAM J. Optim.* **7**(9), 86–125 (1997)
31. Lobo, M.S., Vandenberghe, L., Boyd, S., Lebret, H.: Applications of second order cone programming. *Linear Algebra Appl.* **284**, 193–228 (1998)
32. Monteiro, R.D.C., Adler, I.: Interior path following primal-dual algorithms. Part I: Linear programming. *Math. Programming* **44**, 27–41 (1989)
33. McCrimmon, K.: A taste of Jordan algebras. Springer-Verlag (2003)
34. Monteiro, R.D.C.: Primal-dual path-following algorithms for semidefinite programming. *SIAM J. Optim.* **7**, 663–678 (1997)
35. Monteiro, R.D.C.: Polynomial Convergence of Primal-Dual Algorithms for Semidefinite Programming Based on Monteiro and Zhang Family of Directions. *SIAM J. Optim.* **8**, 797–812 (1998)
36. Monteiro, R.D.C., Tsuchiya, T.: Polynomial convergence of primal-dual algorithms for the second-order cone program based on the MZ-family of directions. *Mathematical Programming* **88**, 61–83 (2000)
37. Monteiro, R.D.C., Zhang, Y.: A unified analysis for a class of path-following primal-dual interior-point algorithms for semidefinite programming. *Mathematical Programming* **81**, 281–299 (1998)
38. Nesterov, Y.: Squared functional systems and optimization problems. In: Frenk, H., Roos, K., Terlaky, T., Zhang, S. (eds.) *High Performance Optimization. Appl. Optim.*, pp. 405–440. Kluwer Acad. Publ., Dordrecht (2000)
39. Nesterov, Y.: Smoothing technique and its application in semidefinite optimization. *Math. Programming* **103**(1), 127–152 (2005)
40. Nesterov, Y., Nemirovski, A.: *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia (1994)

41. Noyan, N., Rudolf, G., Papp, D., Alizadeh, F.: Bilinear Optimality Constraints For the Cone of Positive Polynomials. *Math. Programming* **129**(1), 5–31 (2011)
42. Nemirovski, A., Scheinberg, K.: Extension of Karmarkar’s algorithm onto convex quadratically constrained quadratic programming. *Math. Programming* **72**, 273–289 (1996)
43. Nesterov, Y.E., Todd, M.J.: Self-scaled barriers and interior-point methods for convex programming. *Math. of Oper. Res.* **22**, 1–42 (1997)
44. Nesterov, Y.E., Todd, M.J.: Primal-dual interior-point methods for self-scaled cones. *SIAM J. Optim.* **8**, 324–364 (1998)
45. Pataki, G.: Cone LP’s and Semidefinite Programs: Geometry and a Simplex-type Method. Proceedings of the fifth Integer Programming and Combinatorial Optimization (IPCO) conference (1996)
46. Papp, D., Collado, R., Alizadeh, F.: Extension of the semidefinite characterization of sum of squares functional systems to algebraic structures. Tech. Report 17-10, Rutgers Center for Operations Research (2010)
47. Schmieta, S.H., Alizadeh, F.: Associative and Jordan Algebras, and Polynomial Time Interior-Point Algorithms for Symmetric Cones. *Math. of Oper. Res.* **26**(3), 543–564 (2001)
48. Schmieta, S.H., Alizadeh, F.: Extension of Commutative Class of Primal-Dual Interior Point Algorithms to Symmetric Cones. *Math. Programming* **96**, 409–438 (2003)
49. Schafer, R.D.: An Introduction to Nonassociative Algebras. Academic Press, New York (1966)
50. Schmieta, S.H.: Application of Jordan algebras to the design and analysis of interior-point algorithms for linear, quadratically constrained quadratic, and semi-definite programming. Ph.D. thesis, Rutgers Center for Operations Research, Rutgers University (1999)
51. Springer, T.A.: Jordan Algebras and Algebraic Groups. Springer-Verlag (1973)
52. Tsuchiya, T.: A convergence analysis of the scaling-invariant primal-dual path-following algorithms for second-order cone programming. *Optim. Methods Softw.* **11**, 141–182 (1999)
53. Vinberg, E.: The Theory of Homogeneous Cones. *Transactions of the Moscow Mathematical Society* **12**, 340–403 (1963)
54. Xia, Y., Alizadeh, F.: The  $Q$  method for the Second Order Cone Programming. *Computers and Oper. Research* **35**, 1510–1538 (2008)
55. Xia, Y., Alizadeh, F.: The  $Q$  Method for Symmetric Cone Programming. *Journal of Optimization Theory and Applications* **149**(1), 102–137 (2011)
56. Ye, Y.: A class of projective transformations for linear programming. *SIAM J. Comput.* **19**(3), 457–466 (1990)
57. Ye, Y.: An  $O(n^3 L)$  Potential Reduction Algorithm for Linear Programming. *Math. Programming* **50**(2), 239–258 (1991)
58. Zhang, Y.: On extending primal-dual interior-point algorithms from linear programming to semidefinite programming. *SIAM J. Optim.* **8**, 356–386 (1998)

# Chapter 12

## Complementarity Problems Over Symmetric Cones: A Survey of Recent Developments in Several Aspects

Akiko Yoshise

### 12.1 Introduction

The complementarity problem over a symmetric cone (that we call the *Symmetric Cone Complementarity Problem*, or the *SCCP*) has received much attention of researchers in the last decade. In this chapter, we will provide a brief survey on the recent developments related to the problem.

Let  $\mathcal{V}$  be a finite dimensional real vector space with an inner product denoted by  $\langle \cdot, \cdot \rangle$  and  $\mathcal{K}$  be a symmetric cone in  $\mathcal{V}$  which is a closed convex cone with nonempty interior and self-dual, i.e., satisfies

$$\mathcal{K} = \mathcal{K}^* := \{x \in \mathcal{V} \mid \langle x, y \rangle \geq 0 \text{ for all } y \in \mathcal{K}\}. \quad (12.1)$$

A detailed definition of the symmetric cone will be given in Sect. 12.2.

A typical SCCP is the following *standard SCCP* of the form

$$\left| \begin{array}{l} \text{Find } (x, y) \in \mathcal{K} \times \mathcal{K} \\ \text{s.t. } y - \psi(x) = 0, x \circ y = 0 \end{array} \right. \quad (12.2)$$

where  $\psi : \mathcal{Q} \rightarrow \mathcal{V}$ ,  $\mathcal{Q}$  is an open domain containing  $\mathcal{K}$  and  $\psi$  is differentiable on  $\mathcal{Q}$ . When the function  $\psi$  is affine, we call the problem the *standard linear SCCP*. Many studies have focused on more general problem, the *implicit SCCP*, of the form

$$\left| \begin{array}{l} \text{Find } (x, y, z) \in \mathcal{K} \times \mathcal{K} \times \mathcal{R}^m \\ \text{s.t. } F(x, y, z) = 0, \langle x, y \rangle = 0 \end{array} \right. \quad (12.3)$$

---

A. Yoshise (✉)

Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan

e-mail: [yoshise@sk.tsukuba.ac.jp](mailto:yoshise@sk.tsukuba.ac.jp)

where  $F : \mathcal{Q} \times \mathbb{R}^m \rightarrow \mathcal{V} \times \mathbb{R}^m$ ,  $\mathcal{Q}$  in an open domain containing  $\mathcal{K} \times \mathcal{K} \times \mathbb{R}^m$  and  $F$  is differentiable on  $\mathcal{Q}$ . When the function  $F$  is affine, we call the problem the *implicit linear SCCP* given by

$$\left| \begin{array}{l} \text{Find } (x, y, z) \in \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m \\ \text{s.t. } Px + Qy + Rz - a = 0, \langle x, y \rangle = 0 \end{array} \right. \quad (12.4)$$

where  $a$  is a vector in  $\mathcal{V} \times \mathbb{R}^m$ ,  $P : \mathcal{V} \rightarrow \mathcal{V} \times \mathbb{R}^m$ ,  $Q : \mathcal{V} \rightarrow \mathcal{V} \times \mathbb{R}^m$  and  $R : \mathbb{R}^m \rightarrow \mathcal{V} \times \mathbb{R}^m$  are linear operators. Another important special case of the implicit SCCP is so called the *vertical SCCP* of the form

$$\left| \begin{array}{l} \text{Find } x \in \mathcal{V} \\ \text{s.t. } F(x) \in \mathcal{K}, G(x) \in \mathcal{K}, \langle F(x), G(x) \rangle = 0 \end{array} \right. \quad (12.5)$$

where  $F : \mathcal{V} \rightarrow \mathcal{V}$  and  $G : \mathcal{V} \rightarrow \mathcal{V}$  are differentiable. The vertical SCCP includes the standard SCCP as a special case.

We often assume that the functions associated with the above problems to be monotone. For the implicit SCCP (12.3), if the function  $F$  satisfies

$$\left. \begin{array}{l} (x, y, z), (x', y', z') \in \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m, \\ F(x, y, z) = F(x', y', z') \end{array} \right\} \implies \langle x - x', y - y' \rangle \geq 0 \quad (12.6)$$

it is said to be monotone and we call the problem the *monotone implicit SCCP*. The monotone property (12.6) implies

$$x, x' \in \mathcal{K} \implies \langle F(x) - F(x'), G(x) - G(x') \rangle \geq 0. \quad (12.7)$$

for the functions  $F$  and  $G$  of the vertical SCCP (12.5),

$$x, x' \in \mathcal{K} \implies \langle x - x', \psi(x) - \psi(x') \rangle \geq 0. \quad (12.8)$$

for  $\psi$  of the standard SCCP (12.2), and

$$Px + Qy + Rz = 0 \implies \langle x, y \rangle \geq 0 \quad (12.9)$$

for  $P$ ,  $Q$  and  $R$  of the implicit linear SCCP (12.4).

Note that the monotone implicit linear SCCP (12.4) is a generalization of linear optimization problems over symmetric cones. Consider a primal-dual pair of linear optimization problems over a symmetric cone defined by

$$(P) \min \langle c, x \rangle, \text{ s.t. } Ax = b, x \in \mathcal{K},$$

$$(D) \max b^T z, \text{ s.t. } A^T z + y = c, y \in \mathcal{K}$$

where  $A : \mathcal{V} \rightarrow \Re^m$  is a linear operator,  $b \in \Re^m$  and  $c \in \mathcal{V}$ . Let  $\alpha_P$  and  $\alpha_D$  denote the primal and dual optimal objective values, respectively, i.e.,

$$\begin{aligned}\alpha_P &:= \inf\{\langle c, x \rangle \mid Ax = b, x \in \mathcal{K}\}, \\ \alpha_D &:= \sup\{b^T z \mid A^T z + y = c, y \in \mathcal{K}\}.\end{aligned}$$

It is known that the following duality theorem holds for the problems (P) and (D) (see Theorems 3.2.6 and 3.2.8 of [102]).

**Theorem 12.1 (Duality theorem of the conic optimization).** *If the dual problem (D) is strongly feasible (i.e., there exists  $a(y, z) \in \Re^m \times \mathcal{V}$  such that  $A^T z + y = c$  and  $y \in \text{int } K$ ) and the primal problem (P) is feasible, then the primal problem (P) has an optimal solution. Similarly, if the primal problem (P) is strongly feasible (i.e., there exists an  $x \in \mathcal{V}$  such that  $Ax = b$  and  $x \in \text{int } K$ ) and the dual problem (D) is feasible, then the dual problem (D) has an optimal solution. In either case,  $\alpha_P = \alpha_D$ .*

For any primal feasible solution  $x$  of (P) and any dual feasible solution  $(y, z)$  of (D), we see that

$$\langle c, x \rangle - b^T z = \langle A^T z + y, x \rangle - (Ax)^T z = \langle y, x \rangle \geq 0$$

where the last inequality follows from  $x \in \mathcal{K}$ ,  $y \in \mathcal{K}$  and the self-duality (12.1) of  $\mathcal{K}$ . Therefore, if we define

$$P := \begin{pmatrix} O \\ A \end{pmatrix}, \quad Q := \begin{pmatrix} I \\ O \end{pmatrix}, \quad R := \begin{pmatrix} A^T \\ O \end{pmatrix}, \quad a := \begin{pmatrix} c \\ b \end{pmatrix},$$

then  $P$ ,  $Q$  and  $R$  satisfy

$$Px + Qy + Rz = 0 \implies \langle x, y \rangle = 0$$

which implies that  $Px + Qy + Rz$  is monotone and the corresponding monotone implicit linear SCCP (12.4) is the problem to find an optimal solution of the primal-dual optimization problems over the symmetric cone.

In this chapter, a brief survey of the recent developments on the SCCP will be provided focusing on the following three aspects:

- Interior point methods for the SCCP.
- Merit or smoothing function methods for the SCCP.
- Properties of the SCCP.

After giving a brief introduction to Euclidean Jordan algebras in Sect. 12.2, we review some studies placed in the above three categories, respectively, in Sects. 12.3, 12.4 and 12.5. We will give some concluding remarks in Sect. 12.6.

Before closing this section, we explain some symbols which are used in this chapter. For a given set  $\mathcal{S} \subseteq \mathcal{V}$ ,  $\text{int}\mathcal{S}$  and  $\text{conv}\mathcal{S}$  denote the interior and the convex hull of  $\mathcal{S}$ , respectively.

## 12.2 Euclidean Jordan Algebra

In this section, we give a brief introduction to Euclidean Jordan algebras. See Chap. 11 of this Handbook or the monograph by Faraut and Korányi [27] for a more comprehensive introduction.

A Euclidean Jordan algebra is a triple  $(\mathcal{V}, \circ, \langle \cdot, \cdot \rangle)$  where  $(\mathcal{V}, \langle \cdot, \cdot \rangle)$  is an  $n$ -dimensional inner product space over  $\mathbb{R}$  and  $(x, y) \mapsto x \circ y$  on  $\mathcal{V}$  is a bilinear mapping which satisfies the following conditions for all  $x, y, z \in \mathcal{V}$ :

$$\left\{ \begin{array}{l} \text{(i)} \quad x \circ y = y \circ x, \\ \text{(ii)} \quad x \circ (x^2 \circ y) = x^2 \circ (x \circ y) \text{ where } x^2 = x \circ x, \\ \text{(iii)} \quad \langle x \circ y, z \rangle = \langle x, y \circ z \rangle. \end{array} \right. \quad (12.10)$$

We call  $x \circ y$  the *Jordan product* of  $x$  and  $y$ . Note that  $(x \circ y) \circ w \neq x \circ (y \circ w)$  in general. We assume that there exists an element  $e$  (called as the *identity element*) such that  $x \circ e = e \circ x = x$  for all  $x \in \mathcal{V}$ .

The *rank* of  $(\mathcal{V}, \circ, \langle \cdot, \cdot \rangle)$  is defined as

$$r := \max\{\deg(x) \mid x \in \mathcal{V}\}$$

where  $\deg(x)$  is the degree of  $x \in \mathcal{V}$  given by

$$\deg(x) := \min\{k \mid \{e, x, x^2, \dots, x^k\} \text{ are linearly dependent}\}.$$

The *symmetric cone*  $\mathcal{K}$  is a self-dual (i.e.,  $\mathcal{K}$  satisfies (12.1)) closed convex cone with nonempty interior and homogeneous (i.e., for all  $x, y \in \text{int}\mathcal{K}$ , there exists a invertible linear map  $G$  which satisfies  $G(x) = y$  and  $G(\mathcal{K}) = \mathcal{K}$ ). By Theorem III 2.1 in [27], the symmetric cone  $\mathcal{K}$  coincides with the *set of squares*  $\{x^2 \mid x \in \mathcal{V}\}$  of some Euclidean Jordan algebra  $\mathcal{V}$ .

For any  $x \in \mathcal{V}$ , the *Lyapunov transformation*  $L_x : \mathcal{V} \rightarrow \mathcal{V}$  is defined as  $L_x y = x \circ y$  for all  $y \in \mathcal{V}$ . It follows from (i) and (iii) of (12.10) that the Lyapunov transformation is symmetric, i.e.,  $\langle L_x y, z \rangle = \langle y, L_x z \rangle$  holds for all  $y, z \in \mathcal{V}$ . Especially,  $L_x e = x$  and  $L_x x = x^2$  hold for all  $x \in \mathcal{V}$ . Using the Lyapunov transformation, the *quadratic representation* of  $x \in \mathcal{V}$  is defined as

$$Q_x = 2L_x^2 - L_{x^2}. \quad (12.11)$$

For any  $x \in \mathcal{K}$ ,  $L_x$  is positive semidefinite and it holds that

$$\langle x, y \rangle = 0 \iff x \circ y = 0 \quad (12.12)$$

for any  $x, y \in \mathcal{K}$  (see Lemma 8.3.5 of [1]).

An element  $c \in \mathcal{V}$  is an *idempotent* if  $c^2 = c \neq 0$ , which is also *primitive* if it cannot be written as a sum of two idempotents. A *complete system of orthogonal idempotents* is a finite set  $\{c_1, c_2, \dots, c_k\}$  of idempotents where  $c_i \circ c_j = 0$  for all  $i \neq j$ , and  $c_1 + c_2 + \dots + c_k = e$ . A *Jordan frame* is a complete system of orthogonal primitive idempotents in  $\mathcal{V}$ . The following theorem gives us a *spectral decomposition* for the elements in a Euclidean Jordan algebra (see Theorem III.1.2 of [27]).

**Theorem 12.2 (Spectral decomposition theorem).** *Let  $(\mathcal{V}, \circ, \langle \cdot, \cdot \rangle)$  be a Euclidean Jordan algebra with rank  $r$ . Then for any  $x \in \mathcal{V}$ , there exist a Jordan frame  $\{c_1, c_2, \dots, c_r\}$  and real numbers  $\lambda_i(x)$  ( $i = 1, 2, \dots, r$ ) such that*

$$x = \sum_{i=1}^r \lambda_i(x) c_i.$$

The numbers  $\lambda_i(x)$  ( $i = 1, 2, \dots, r$ ) are called the *eigenvalues* of  $x$ , which are uniquely determined by  $x$ .

Note that the Jordan frame in the above theorem depends on  $x$ , but we omit the dependence in order to simplify the notation. If two elements  $x$  and  $y$  share the same Jordan frames in the decompositions in Theorem 12.2 then they *operator commute*, i.e., they satisfy  $L_x L_y = L_y L_x$ . An eigenvalue  $\lambda_i(x)$  is continuous with respect to  $x$ . The *trace* of  $x$  is defined as  $\text{tr}(x) = \sum_{i=1}^r \lambda_i(x)$  and the *determinant* of  $x$  is defined as  $\det(x) = \prod_{i=1}^r \lambda_i(x)$ . We also see that  $x \in \mathcal{K}$  (respectively,  $x \in \text{int}\mathcal{K}$ ) if and only if  $\lambda_i(x) \geq 0$  (respectively,  $\lambda_i(x) > 0$ ) for all  $i = 1, 2, \dots, r$ . For any  $x \in \mathcal{V}$  having the spectral decomposition  $x = \sum_{i=1}^r \lambda_i(x) c_i$ , we denote

$$x^{1/2} := \sqrt{x} := \sum_{i=1}^r \sqrt{\lambda_i(x)} c_i \quad \text{if } \lambda_i(x) \geq 0 \text{ for all } i = 1, 2, \dots, r,$$

$$x^{-1} := \sum_{i=1}^r \lambda_i(x)^{-1} c_i \quad \text{if } \lambda_i(x) \neq 0 \text{ for all } i = 1, 2, \dots, r.$$

Now we introduce another decomposition, the *Peirce decomposition*, on the space  $\mathcal{V}$  (see Theorem IV.2.1 of [27]).

**Theorem 12.3 (Peirce decomposition theorem).** *Let  $\{c_1, \dots, c_r\}$  be a Jordan frame. Define*

$$\mathcal{V}_i := \{\theta c_i \mid \theta \in \mathfrak{R}\},$$

$$\mathcal{V}_{ij} := \left\{ x \in \mathcal{V} \mid c_i \circ x = \frac{1}{2}x = c_j \circ x \right\} \quad (i < j).$$

Then for any  $x \in \mathcal{V}$ , there exists  $x_i \in \mathfrak{R}$ ,  $c_i \in \mathcal{V}_i$  and  $x_{ij} \in \mathcal{V}_{ij}$  ( $i < j$ ),

$$x = \sum_{i=1}^r x_i c_i + \sum_{i < j} x_{ij}.$$

Fix a Jordan frame  $\{c_1, c_2, \dots, c_l\}$  and define

$$\mathcal{V}^{(l)} := \{x \in \mathcal{V} \mid x \circ (c_1 + c_2 + \dots + c_l) = x\}$$

for  $1 \leq l \leq r$ . Corresponding to  $\mathcal{V}^{(l)}$ , we consider the (orthogonal) projection  $P^{(l)} : \mathcal{V} \rightarrow \mathcal{V}^{(l)}$ . For a given linear transformation  $L : \mathcal{V} \rightarrow \mathcal{V}$ , we denote  $L_{\{c_1, c_2, \dots, c_l\}}$  the composite transformation  $P^{(l)} \bullet L : \mathcal{V} \rightarrow \mathcal{V}^{(l)}$ . We call  $L_{\{c_1, c_2, \dots, c_l\}}$  the *principal subtransformation* of  $L$  corresponding to  $\{c_1, c_2, \dots, c_l\}$  and the determinant of  $L_{\{c_1, c_2, \dots, c_l\}}$  a *principal minor* of  $L$ .

Typical examples of Euclidean Jordan algebras are

(i) *Euclidean Jordan algebra of  $n$ -dimensional vectors* where

$$\mathcal{V} = \mathfrak{R}^n, \quad \mathcal{K} = \mathfrak{R}_+^n := \{x \in \mathfrak{R}^n \mid x \geq 0\},$$

(ii) *Euclidean Jordan algebra of  $n$ -dimensional symmetric matrices* where

$$\mathcal{V} = \mathcal{S}^n := \{X \in \mathfrak{R}^{n \times n} \mid X = X^T\}, \quad \mathcal{K} = \mathcal{S}_+^n := \{X \in \mathcal{S}^n \mid X \succeq O\}$$

and  $X \succeq O$  denotes that  $X$  is a positive semidefinite matrix, and

(iii) *Euclidean Jordan algebra of quadratic forms* where

$$\mathcal{V} = \mathfrak{R}^n, \quad \mathcal{K} = \mathcal{L}^n := \{(x_1, x_2) \in \mathfrak{R} \times \mathfrak{R}^{n-1} \mid \|x_2\| \leq x_1\}$$

and  $\|\cdot\|$  denotes the Euclidean norm.

In this chapter, we call the SCCP the *nonlinear complementarity problem* (NCP) when  $\mathcal{V}$  and  $\mathcal{K}$  are given in (i), the *semidefinite complementarity problem* (SDCP) when  $\mathcal{V}$  and  $\mathcal{K}$  are given in (ii) and the *second order cone complementarity problem* (SOCCP) when  $\mathcal{V}$  and  $\mathcal{K}$  are given in (iii), respectively.

## 12.3 Interior-Point Methods for the SCCP

The first interior point algorithm for solving the SCCP has been provided in Chap. 7 of Nesterov and Nemirovski's seminal book [85] while the connection to the Euclidean Jordan algebra has not been pointed out clearly. The algorithm is an interior point algorithm based on a *self-concordant barrier* only in the variables  $x$ , which is closely related to symmetric cones [42]. The polynomial complexity bound and a way to find an appropriate initial point have been discussed.

After about four years from the publication of [85], in 1997, a first interior point algorithm in the setting of a Euclidean Jordan algebra has been given by Faybusovich [28] which employs the barrier function in  $x$  and  $y$  of the form

$$\mu\langle x, y \rangle - \log \det(x) - \log \det(y).$$

Independently of Faybusovich's work, an interior point algorithm for solving the  $n$ -dimensional monotone implicit linear SDCP (12.4) (called as the *monotone implicit SDLCP*) has been provided by Kojima et al. [52]. In the paper, after showing that the convex quadratic semidefinite optimization problem can be cast both into a monotone SDLCP and into a semidefinite optimization problem (called as the *SDP*), the authors have mentioned that

This fact itself never denies the significance of the monotone SDLCP because the direct SDLCP formulation is of a smaller size than the SDP formulation but raises questions like how general the monotone SDLCP is and whether it is essentially different from the semidefinite optimization problem. In their recent paper [53], Kojima, Shida and Shindoh showed that the monotone SDLCP is reducible to a SDP involving an additional  $m$ -dimensional variable vector and an  $(m+1) \times (m+1)$  variable symmetric matrix, where  $m = n(n+1)/2$ .

The convex cone  $\mathcal{K}$  considered in [53] is quite general, i.e.,  $\mathcal{K}$  is just nonempty closed and convex. Therefore, the raised questions in [52] are also the questions to the monotone implicit linear SCCP (12.4).

In fact, many results on the primal-dual interior point algorithms for solving the SDP can be easily extended to solve the monotone implicit linear SDCP (12.4) (i.e., monotone implicit SDLCP), and then the extended results can be further extended to solve the monotone implicit linear SCCP (12.4) using the fundamental results established by Alizadeh and Schmieta [1, 104] for the primal and dual symmetric cone linear optimization problems.

This may be a reason why there are far fewer papers on interior point algorithms for solving the monotone implicit linear SCCP (12.4) than those for solving the primal and dual symmetric cone linear optimization problems.

Let us consider a more general problem, the monotone implicit SCCP (12.3) satisfying (12.6). In view of the property (12.12), the problem (12.3) is equivalent to find a  $(x, y, z) \in \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m$ , i.e., it satisfies

$$x \circ y = 0, \quad F(x, y, z) = 0. \quad (12.13)$$

Any interior point algorithm for solving the problem generates a sequence  $\{(x_k, y_k, z_k)\} \subset \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m$  satisfying  $x_k \in \text{int}\mathcal{K}$  and  $y_k \in \text{int}\mathcal{K}$  ( $k = 1, 2, \dots$ ). Let  $(x_0, y_0, z_0) \in \text{int}\mathcal{K} \times \text{int}\mathcal{K} \times \mathbb{R}^m$  be an arbitrary initial point. It is not too often that the initial point  $(x_0, y_0, z_0)$  is a *feasible interior point* of (12.3) such that  $F(x_0, y_0, z_0) = 0$ . So in practice, we have to assume that  $F(x_0, y_0, z_0) \neq 0$ . Obviously, the system

$$x \circ y = x_0 \circ y_0, \quad F(x, y, z) = F(x_0, y_0, z_0) \quad (12.14)$$

has a trivial solution  $(x_0, y_0, z_0)$  while the target system is given by (12.13). Let  $H : \text{int}\mathcal{K} \times \text{int}\mathcal{K} \times \mathbb{R}^m \rightarrow \mathcal{V} \times \mathcal{V} \times \mathbb{R}^m$  be the so-called *interior point map* which is defined as

$$H(x, y, z) := (x \circ y, F(x, y, z)). \quad (12.15)$$

Then the systems (12.14) and (12.13) can be represented using  $H$  as

$$H(x, y, z) = (x_0 \circ y_0, F(x_0, y_0, z_0)) \quad (12.16)$$

and

$$H(x, y, z) = (0, 0), \quad (12.17)$$

respectively. Introducing a parameter  $\mu \in (0, 1]$ , consider the system

$$H(x, y, z) = \mu(x_0 \circ y_0, F(x_0, y_0, z_0)). \quad (12.18)$$

The system has a trivial solution when  $\mu = 1$ . If the system (12.18) has the unique solution  $(x(\mu), y(\mu), z(\mu))$  for each  $\mu \in (0, 1]$  and  $(x(\mu), y(\mu), z(\mu))$  is continuous at  $\mu \in (0, 1)$ , then we may numerically trace the path  $\{(x(\mu), y(\mu)), z(\mu)\}$  from the initial point  $(x_0, y_0, z_0)$  to a solution of the monotone implicit SCCP (12.3). This is a basic idea of the (infeasible) interior point algorithm for the SCCP.

Most of studies related to the interior point algorithms for the SCCP have dealt with one of the following subjects:

- The interior point map and its properties.
- Algorithms and their convergence properties.

In what follows, we observe some recent results on the above subjects.

### 12.3.1 Interior Point Map and Its Properties

It is important to observe the properties of the interior point map  $H$  of (12.15) to find whether the system (12.18) has the unique solution  $(x(\mu), y(\mu), z(\mu))$  for each  $\mu \in (0, 1]$ . If it holds then the set  $\{(x(\mu), y(\mu), z(\mu)) \mid \mu \in (0, 1]\}$  will give us an (*infeasible*) *interior point trajectory* of the SCCP.

For the monotone SDCP, Shida et al. [105] investigated the existence and continuity of the (infeasible) interior point trajectory in a general setting of the problem. On the other hand, Monteiro [81] showed a vast set of conclusions concerning the interior point map for the monotone SDCP. Based on the paper [81], the following results have been shown in [124] for the monotone implicit SCCP (12.3) satisfying (12.6).

Let  $\mathcal{U}$  be a subset of  $\text{int } \mathcal{K} \times \text{int } \mathcal{K}$  defined by

$$\mathcal{U} := \{(x, y) \in \text{int } \mathcal{K} \times \text{int } \mathcal{K} : x \circ y \in \text{int } \mathcal{K}\}.$$

The following assumption has been imposed in [124].

- Assumption 12.1**
- (i)  $F : \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m \rightarrow \mathcal{V} \times \mathbb{R}^m$  is monotone on its domain, i.e.,  $F$  satisfies (12.6).
  - (ii)  $F : \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m \rightarrow \mathcal{V} \times \mathbb{R}^m$  is  $z$ -bounded on its domain, i.e., for any sequence  $\{(x_k, y_k, z_k)\}$  in the domain of  $F$ , if  $\{(x_k, y_k)\}$  and  $\{F(x_k, y_k, z_k)\}$  are bounded then the sequence  $\{z_k\}$  is also bounded.
  - (iii)  $F : \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m \rightarrow \mathcal{V} \times \mathbb{R}^m$  is  $z$ -injective on its domain, i.e., if  $(x, y, z)$  and  $(x, y, z')$  lie in the domain of  $F$  and satisfy  $F(x, y, z) = F(x, y, z')$ , then  $z = z'$  holds.

Under the assumption above, the homeomorphism of the interior point map  $H$  has been shown (see Theorem 3.10 of [124]).

**Theorem 12.4 (Homeomorphism of the interior point map).** Suppose that a continuous map  $F : \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m \rightarrow \mathcal{V} \times \mathbb{R}^m$  satisfies Assumption 12.1. Then the map  $H$  defined by (12.15) maps  $\mathcal{U} \times \mathbb{R}^m$  homeomorphically onto  $\mathcal{K} \times F(\mathcal{U} \times \mathbb{R}^m)$ .

The theorem above ensures that if the monotone implicit SCCP (12.3) has an interior feasible point  $(\bar{x}, \bar{y}, \bar{z}) \in \text{int } \mathcal{K} \times \text{int } \mathcal{K} \times \mathbb{R}^m$  which satisfies

$$\bar{x} \circ \bar{y} \in \text{int } \mathcal{K} \text{ and } F(\bar{x}, \bar{y}, \bar{z}) = 0$$

and if we have a bounded path  $\{p(\mu) \mid \mu \in [0, 1]\} \subseteq \mathcal{K} \times F(\mathcal{U} \cap \mathbb{R}^m)$  such that

$$p(0) = 0 \text{ and } p(\mu) \in \text{int } \mathcal{K} \times F(\mathcal{U} \times \mathbb{R}^m)$$

then there exists a unique path  $\{(x(\mu), y(\mu), z(\mu)) \mid \mu \in (0, 1]\} \subseteq \text{int } \mathcal{K} \times \text{int } \mathcal{K} \times \mathbb{R}^m$  for which

$$H(x(\mu), y(\mu), z(\mu)) = p(\mu) \text{ for all } \mu \in (0, 1]$$

holds and whose any accumulation point is a solution of the monotone implicit SCCP (12.3). Thus the path  $\{(x(\mu), y(\mu), z(\mu)) \mid \mu \in (0, 1]\}$  is an (*infeasible*) *interior point trajectory* (see Corollary 4.4 of [124]). A condition on  $F$  so that we can take

$$p(\mu) = \mu(x_0 \circ y_0, F(x_0, y_0, z_0))$$

as in (12.18) has been provided in [124]. Note that if  $x_0 \circ y_0 = e$ , we sometimes call the (*infeasible*) interior point trajectory the (*infeasible*) *central trajectory* or (*infeasible*) *central path*.

### 12.3.2 Algorithms and Their Convergence Properties

In what follows, we will describe outlines of several interior point algorithms which are based on tracing the (infeasible) interior point trajectory  $\{(x(\mu), y(\mu), z(\mu)) \mid \mu \in (0, 1]\}$ .

#### 12.3.2.1 Infeasible Interior Point Algorithm

Potra [98] proposed an infeasible interior point algorithm for solving the monotone implicit linear SCCP (12.4). The algorithm is a generalization of the *corrector-predictor approach* proposed in [40, 97]. The outline of the algorithm is as follows.

Let  $(\mu_0, x_0, y_0, z_0)$  be a starting point satisfying  $\mu_0 = 1$  and  $x_0 \circ y_0 = e$ . At each iteration  $(x, y, z)$ , the target system is given by

$$H(x, y, z) = \mu_k (x_0 \circ y_0, 0) = \mu_k (e, 0) \quad (12.19)$$

where  $\mu_k \in (0, \mu_0]$ . Applying Newton's method to the system (12.19) at  $(x, y, z)$  leads us to the linear system

$$\begin{aligned} y \circ \Delta x + \Delta y \circ x &= \gamma \mu_k e - x \circ y, \\ P \Delta x + Q \Delta y + R \Delta z &= (1 - \gamma)(a - Px - Qy - Rz) \end{aligned} \quad (12.20)$$

where  $\gamma \in [0, 1]$  is a parameter for regulating the feasibility and the complementarity.

Let us choose a  $p \in \text{int } \mathcal{K}$  belonging to the *commutative class of scalings* for  $(x, y)$ ,

$$C(x, y) = \{p \in \text{int } \mathcal{K} \mid Q_p x \text{ and } Q_{p^{-1}} y \text{ operator commute}\} \quad (12.21)$$

where  $Q_p$  is the quadratic representation of  $p$  defined by (12.11), and consider the scaled quantities

$$\tilde{x} = Q_p x, \tilde{y} = Q_{p^{-1}} y, \tilde{P} = PQ_{p^{-1}}, \tilde{Q} = QQ_p.$$

Since  $\tilde{x} \circ \tilde{y} = \mu_k e$  if and only if  $x \circ y = \mu_k e$ , the target system (12.19) can be written under the form

$$\tilde{x} \circ \tilde{y} = \mu_k e, \quad \tilde{P} \tilde{x} + \tilde{Q} \tilde{y} + Rz = a$$

and the Newton system at  $(\tilde{x}, \tilde{y}, z)$  is given by

$$\begin{aligned} \tilde{y} \circ \tilde{\Delta} x + \tilde{\Delta} y \circ \tilde{x} &= \gamma \mu_k e - \tilde{x} \circ \tilde{y}, \\ \tilde{P} \tilde{\Delta} x + \tilde{Q} \tilde{\Delta} y + R \tilde{\Delta} z &= (1 - \gamma)(a - \tilde{P} \tilde{x} - \tilde{Q} \tilde{y} - Rz). \end{aligned}$$

Since we have chosen  $p$  to be in the commutative class of scalings  $C(x, y)$ , the resulting  $\tilde{x}$  and  $\tilde{y}$  share the same Jordan frame, the above system has a unique

solution  $(\tilde{\Delta}x, \tilde{\Delta}y, \tilde{\Delta}z)$  and we obtain a *commutative class of the search directions*  $(\Delta x, \Delta y, \Delta z)$  by

$$(\Delta x, \Delta y, \Delta z) := (Q_p^{-1} \tilde{\Delta}x, Q_{p^{-1}}^{-1} \tilde{\Delta}y, \Delta z)$$

which satisfies (12.20). The commutative class of the search directions is a subclass of Monteiro and Zhang family (see [82, 83]) and well used in many interior point algorithms for solving optimization problems over the symmetric cones.

The algorithm employs the following *neighborhood of the infeasible central path*

$$\begin{aligned} \mathcal{N}(\beta) = & \{(\mu, x, y, z) \in (0, \mu_0] \times \text{int } \mathcal{K} \times \text{int } \mathcal{K} \\ & \times \mathfrak{R}^m \mid \sigma(x, y, z) \subset [\beta\mu, (1/\beta)\mu]\} \end{aligned} \quad (12.22)$$

where  $\beta \in (0, 1)$  is a given parameter, and  $\sigma(x, y, z)$  denotes the set of all eigenvalues of  $Q_{x^{1/2}y}$  with  $Q_{x^{1/2}}$  being the quadratic representation of  $x^{1/2}$  (see [101]). Let  $\lambda_{\min}(x, y, z)$  and  $\lambda_{\max}(x, y, z)$  be the minimum and maximum eigenvalues in  $\sigma(x, y, z)$ , respectively. By introducing the proximity measure

$$\delta(\mu, x, y, z) = \max \left\{ 1 - \frac{\lambda_{\min}(x, y, z)}{\mu}, 1 - \frac{\mu}{\lambda_{\max}(x, y, z)} \right\} \quad (12.23)$$

the neighborhood (12.22) can be written as

$$\mathcal{N}(\beta) = \{(\mu, x, y, z) \in (0, \mu_0] \times \text{int } \mathcal{K} \times \text{int } \mathcal{K} \times \mathfrak{R}^m \mid \delta(\mu, x, y, z) \leq 1 - \beta\}.$$

Each iteration of the algorithm consists of two steps, a corrector step and a predictor step.

The intent of the *corrector step* is to increase proximity to the central path. We choose  $\gamma \in [\underline{\gamma}, \bar{\gamma}]$  and  $\eta = 1 - \gamma$  where  $0 < \gamma < \bar{\gamma} < 1$  are given parameters, compute the search directions  $(\Delta x, \Delta y, \Delta z)$  at  $(x, y, z)$ , find

$$\alpha_c := \arg \min \{\delta(\mu, x + \alpha \Delta x, y + \alpha \Delta y, z + \alpha \Delta z)\},$$

set  $(x, y, z) \leftarrow (x, y, z) + \alpha_c (\Delta x, \Delta y, \Delta z)$  and proceed to the predictor step.

In contrast, the intent of the *predictor step* is to decrease the complementarity gap as much as possible while keeping the iterate in  $\mathcal{N}(\beta)$ . We choose  $\gamma = 0$  and  $\eta = 1$ , compute the search directions  $(\Delta x, \Delta y, \Delta z)$  at  $(x, y, z)$ , find

$$\alpha_p := \max \{\bar{\alpha} \mid (x, y, z) + \alpha (\Delta x, \Delta y, \Delta z) \in \mathcal{N}(\beta) \text{ for each } \alpha \in [0, \bar{\alpha}]\},$$

set  $(x, y, z) \leftarrow (x, y, z) + \alpha_p (\Delta x, \Delta y, \Delta z)$  and proceed to the next iterate.

While the detailed proof has not been proposed, the author asserted that the algorithm generates a sequence  $\{(\mu_k, x_k, y_k, z_k)\}$  satisfying  $(x_k, y_k, z_k) \in \mathcal{N}(\beta)$  and the polynomial convergence can be obtained under general assumptions using general theoretical results from [104] and [101]. It has been also asserted that the superlinear

convergence is proved if the problem has a strictly complementary solution and the sequence  $\{(x_k, y_k, \kappa_k)\}$  satisfies  $(x_k \circ y_k) / \sqrt{\langle x_k, y_k \rangle} \rightarrow 0$ .

There are several papers on infeasible interior point algorithms for solving the monotone and linear SDCP, e.g., [52, 95].

### 12.3.2.2 Homogeneous Algorithm

Another approach to overcome the difficulty to find a feasible interior starting point is the homogeneous algorithm proposed in [124] and [125] for solving the monotone standard SCCP (12.2) satisfying (12.8), which is a generalization of the homogeneous algorithm in [2] for the classical symmetric cone  $\mathcal{K} = \mathfrak{R}_+^n$ .

The homogeneous algorithm is an infeasible interior point algorithm for the *homogeneous model* given by

$$\begin{aligned} & \text{(HCP) Find } (x, \tau, y, \kappa) \in (\mathcal{K} \times \mathfrak{R}_{++}) \times (\mathcal{K} \times \mathfrak{R}_+) \\ & \text{s.t. } F_{\text{H}}(x, \tau, y, \kappa) = 0, (x, \tau) \circ_{\text{H}} (y, \kappa) = 0 \end{aligned} \quad (12.24)$$

where  $F_{\text{H}} : (\mathcal{K} \times \mathfrak{R}_{++}) \times (\mathcal{K} \times \mathfrak{R}_+) \rightarrow (\mathcal{V} \times \mathfrak{R})$  and  $(x, \tau) \circ_{\text{H}} (y, \kappa)$  are defined as

$$F_{\text{H}}(x, \tau, y, \kappa) := (y, \kappa) - \psi_{\text{H}}(x, \tau), \quad \psi_{\text{H}}(x, \tau) := (\tau \psi(x/\tau), -\langle \psi(x/\tau), x \rangle) \quad (12.25)$$

and

$$(x, \tau) \circ_{\text{H}} (y, \kappa) := (x \circ y, \tau \kappa). \quad (12.26)$$

The function  $\psi_{\text{H}}$  has the following property (see Proposition 5.3 of [124]).

**Proposition 12.1 (Monotonicity of the homogeneous function  $\psi_{\text{H}}$ ).** *If  $\psi : \mathcal{K} \rightarrow \mathcal{V}$  is monotone, i.e., satisfies (12.8), then the function  $\psi_{\text{H}}$  is monotone on  $\text{int } \mathcal{K} \times \mathfrak{R}_{++}$ .*

For ease of notation, we use the following symbols

$$\mathcal{V}_{\text{H}} := \mathcal{V} \times \mathfrak{R}, \quad \mathcal{K}_{\text{H}} := \mathcal{K} \times \mathfrak{R}_+, \quad x_{\text{H}} := (x, \tau) \in \mathcal{V}_{\text{H}},$$

$$y_{\text{H}} := (y, \kappa) \in \mathcal{V}_{\text{H}}, \quad e_{\text{H}} := (e, 1).$$

Note that the set  $\mathcal{K}_{\text{H}}$  is a Cartesian product of two symmetric cones  $\mathcal{K}$  and  $\mathfrak{R}_+$  given by

$$\mathcal{K}_{\text{H}} = \left\{ x_{\text{H}}^2 = (x^2, \tau^2) : x_{\text{H}} \in \mathcal{V}_{\text{H}} \right\}$$

which implies that  $\mathcal{K}_{\text{H}}$  is the symmetric cone of  $\mathcal{V}_{\text{H}}$ . It can be easily seen that  $\text{int } \mathcal{K}_{\text{H}} = \text{int } \mathcal{K} \times \mathfrak{R}_{++}$ .

A merit of the homogeneous model is that it can provide certifications on strong feasibility or strong infeasibility of the original problem by adding the new variables  $\tau$  and  $\kappa$ . The following theorem has been shown (see Theorem 5.4 of [124]).

**Theorem 12.5 (Properties of the homogeneous model).**

- (i) The HCP (12.24) is asymptotically feasible, i.e., there exists a bounded sequence  $\{(x_{\text{H}}^{(k)}, y_{\text{H}}^{(k)})\} \subseteq \text{int } \mathcal{K}_{\text{H}} \times \text{int } \mathcal{K}_{\text{H}}$  such that

$$\lim_{k \rightarrow \infty} F_{\text{H}}(x_{\text{H}}^{(k)}, y_{\text{H}}^{(k)}) = \lim_{k \rightarrow \infty} (y_{\text{H}} - \psi_{\text{H}}(x_{\text{H}}^{(k)})) = 0.$$

- (ii) The monotone standard SCCP (12.2) has a solution if and only if the HCP (12.24) has an asymptotic solution  $(x_{\text{H}}^*, y_{\text{H}}^*) = (x^*, \tau^*, y^*, \kappa^*)$  with  $\tau^* > 0$ . In this case,  $(x^*/\tau^*, y^*/\tau^*)$  is a solution of the monotone standard SCCP (12.2).  
(iii) Suppose that  $\psi$  satisfies the Lipschitz condition on  $\mathcal{K}$ , i.e., there exists a constant  $\gamma \geq 0$  such that

$$\|\psi(x+h) - \psi(x)\| \leq \gamma \|h\| \text{ for any } x \in \mathcal{K} \text{ and } h \in \mathcal{V} \text{ such that } x+h \in \mathcal{K}.$$

If the monotone standard SCCP (12.2) is strongly infeasible, i.e., there is no sequence  $\{x^{(k)}, y^{(k)}\} \subseteq \text{int } K \times \text{int } K$  such that  $\lim_{k \rightarrow \infty} (y^{(k)} - \psi(x^{(k)})) = 0$ , then the HCP (12.24) has an asymptotic solution  $(x^*, \tau^*, y^*, \kappa^*)$  with  $\kappa^* > 0$ . Conversely, if the HCP (12.24) has an asymptotic solution  $(x^*, \tau^*, y^*, \kappa^*)$  with  $\kappa^* > 0$  then the monotone standard SCCP (12.2) is infeasible. In the latter case,  $(x^*/\kappa^*, y^*/\kappa^*)$  is a certificate to prove infeasibility of the monotone standard SCCP (12.2).

Unfortunately, the homogeneous function  $\psi_{\text{H}}$  undermines linearity of the original function  $\psi$  if it has. Let us consider a simple example,  $\psi : \mathbb{R} \rightarrow \mathbb{R}$ ,  $\psi(x) = x$ . Then the induced homogeneous function  $\psi_{\text{H}} : \mathbb{R} \times \mathbb{R}_{++} \rightarrow \mathbb{R}^2$  is given by  $\psi_{\text{H}}(x, \tau) = (x, -x^2/\tau)$ . The function  $\psi_{\text{H}}$  is no longer linear, but  $\psi_{\text{H}}$  is monotone as in Theorem 12.1 and satisfies the following assumption which can be considered as a scaled Lipschitz condition.

**Assumption 12.2** There exists a  $\theta \geq 0$  such that

$$\left\| \tilde{z}(\alpha) \circ (\tilde{\psi}(\tilde{z}(\alpha)) - \tilde{\psi}(\tilde{z}) - \alpha D\tilde{\psi}(\tilde{z})\Delta z) \right\| \leq \alpha^2 \theta \langle \Delta z, D\tilde{\psi}(\tilde{z})\Delta z \rangle$$

for all  $z \in \text{int } \mathcal{K}$ ,  $\Delta z \in \mathcal{V}$ ,  $p \in C(x, y)$  and  $\alpha \in [0, 1]$  such that  $z(\alpha) \in \text{int } \mathcal{K}$ , where

$$\tilde{z}(\alpha) = Q_p(z + \alpha \Delta z), \quad \tilde{\psi}(\tilde{z}) = Q_p^{-1} \bullet \psi \bullet Q_p^{-1}(\tilde{z}) = Q_p^{-1}\psi(z).$$

Here,  $\phi_1 \bullet \phi_2$  denotes the composite function of  $\phi_1$  and  $\phi_2$ .

Obviously, if  $\psi$  is affine then  $\psi$  satisfies the assumption with  $\theta = 0$ .

The homogeneous algorithm in [125] is an infeasible interior point to the homogeneous model (12.24). It employs the commutative class of search directions (see (12.21)) including the *yx-direction* (respectively, *xy-direction*) with  $p = y^{1/2}$  (respectively,  $p = x^{-1/2}$ ) and the *Nesterov–Todd (NT) direction* with

$$p = \left[ Q_{x^{1/2}}(Q_{x^{1/2}}y)^{-1/2} \right]^{-1/2} = \left[ Q_{y^{-1/2}}(Q_{y^{-1/2}}x)^{1/2} \right]^{-1/2}.$$

so that  $\tilde{x} = \tilde{y}$ , and the following three types of neighborhoods

$$\begin{aligned}\mathcal{N}_F(\beta) &:= \{(x_H, y_H) \in \mathcal{K}_H \times \mathcal{K}_H \mid d_F(x_H, y_H) \leq \beta \mu_H\}, \\ \mathcal{N}_2(\beta) &:= \{(x_H, y_H) \in \mathcal{K}_H \times \mathcal{K}_H \mid d_2(x_H, y_H) \leq \beta \mu_H\}, \\ \mathcal{N}_{-\infty}(\beta) &:= \{(x_H, y_H) \in \mathcal{K}_H \times \mathcal{K}_H \mid d_{-\infty}(x_H, y_H) \leq \beta \mu_H\}\end{aligned}\quad (12.27)$$

where  $\beta \in (0, 1)$ ,  $w_H = Q_{x_H^{1/2}} y_H$  and

$$\begin{aligned}\mu_H &:= \langle x_H, y_H \rangle / (r + 1) \\ d_F(x_H, y_H) &:= \|Q_{x_H^{1/2}} y_H - \mu_H e_H\|_F = \sqrt{\sum_{i=1}^{r+1} (\lambda_i(w_H) - \mu_H)^2} \\ d_2(x_H, y_H) &:= \|Q_{x_H^{1/2}} y_H - \mu_H e_H\|_2 \\ &= \max \{|\lambda_i(w_H) - \mu_H| \mid i = 1, \dots, r+1\} \\ &= \max \{\lambda_{\max}(w_H) - \mu_H, \mu_H - \lambda_{\min}(w_H)\}, \\ d_{-\infty}(x_H, y_H) &:= \mu_H - \lambda_{\min}(w_H).\end{aligned}$$

Since the inclusive relation  $\mathcal{N}_F(\beta) \subseteq \mathcal{N}_2(\beta) \subseteq \mathcal{N}_{-\infty}(\beta)$  holds for any  $\beta \in (0, 1)$  (see Proposition 29 of [104]), we call the algorithms using  $\mathcal{N}_F(\beta)$ ,  $\mathcal{N}_2(\beta)$  and  $\mathcal{N}_{-\infty}(\beta)$  the *short-step algorithm*, the *semi-long-step algorithm* and the *long-step algorithm*, respectively.

Under Assumption 12.2, we obtain the following results by analogous discussions as in [104] and in [101] (see Corollary 7.3 of [124]).

**Theorem 12.6 (Iteration bounds of homogeneous algorithms).** *Suppose that  $\psi : \mathcal{K} \rightarrow \mathcal{V}$  satisfies Assumption 12.2, and we use the NT, xy or yx method for determining the search direction. Then the number of iterations of each homogeneous algorithm is bounded as follows:*

	NT method	xy or yx method
Short-step using $\mathcal{N}_F(\beta)$	$O(\sqrt{r}(1 + \sqrt{r}\theta) \log \epsilon^{-1})$	$O(\sqrt{r}(1 + \sqrt{r}\theta) \log \epsilon^{-1})$
Semi-long-step using $\mathcal{N}_2(\beta)$	$O(r(1 + \sqrt{r}\theta) \log \epsilon^{-1})$	$O(r(1 + \sqrt{r}\theta) \log \epsilon^{-1})$
Long-step using $\mathcal{N}_{-\infty}(\beta)$	$O(r(1 + \sqrt{r}\theta) \log \epsilon^{-1})$	$O(r^{1.5}(1 + \sqrt{r}\theta) \log \epsilon^{-1})$

Here  $\epsilon > 0$  is the accuracy parameter and  $\theta \geq 0$  is the scaled Lipschitz parameter appearing in Assumption 12.2, and  $\theta = 0$  holds if  $\psi$  is affine.

### 12.3.2.3 Other Infeasible Interior Point Algorithms

In [74], another infeasible interior point algorithm has been provided for solving the standard linear SCCP. The algorithm is essentially a combination of the standard interior-point methods and the smoothing Newton methods described in the next section, and can be regarded as an extension of the algorithm proposed by Xu and Burke [121]. The algorithm is based on the Chen–Harker–Kanzow–Smale (CHKS) smoothing function  $\Psi_\mu : \mathcal{V} \times \mathcal{V} \rightarrow \mathcal{V}$  (see Sect. 12.4.1) and the following target system

$$\begin{pmatrix} y - Px - a \\ \Psi_\mu(x, y) \end{pmatrix} = 0 \quad (12.28)$$

is considered with some parameter  $\mu > 0$ . Each iteration employs an approximate Newton method with the Nesterov–Todd direction and the semi-long step neighborhood  $N_2(\beta)$  in (12.27). The algorithm enjoys the following theorem (see Theorem 4.7 of [74]).

**Theorem 12.7 (Iteration bounds of combined algorithms).** *Let  $\epsilon > 0$  be a given accuracy parameter. If the standard linear SCCP has a solution then the algorithm is polynomial-time convergent and the iteration complexity is  $O(\sqrt{r} \log \epsilon^{-1})$  for the feasible version of the algorithm and  $O(r \log \epsilon^{-1})$  for the infeasible version of the algorithm.*

The subsequent iterated point by the combined algorithm automatically remains in the neighborhood of central path for the full Newton step, which is a merit of the algorithm.

### 12.3.2.4 Local Convergence Properties

Very few studies on local convergence properties of interior point algorithms for the SCCP are yet available while a description on this issue can be found in [98]. On the other hand, many studies have been done on the monotone implicit linear SDCP (12.4) (called as the monotone implicit SDLCP).

There are two directions in the study of local convergence of interior point algorithms for the monotone implicit SDLCP (12.4).

The first approach is to analyze the generated sequence by the infeasible interior point algorithms employing several types of search directions and neighborhoods (see, e.g., [54, 55, 73, 95]).

The second approach is to analyze the infeasible interior point trajectory (central path) [99] or a neighboring *off-central path* of the monotone implicit SDLCP (12.4) with  $m = 0$  (i.e., the variable  $z$  has vanished) [106–108] which is defined by the following ordinary differential equations (ODEs)

$$\begin{aligned} H_P(XV + UY) &= \frac{1}{\mu} H_P(XY), \\ P(U) + Q(V) &= O, \\ (X(1), Y(1)) &= (X_0, Y_0) \end{aligned} \tag{12.29}$$

where  $\mu > 0$  is a parameter,  $X$  and  $Y$  are functions of  $\mu$ ,  $U$  and  $V$  are derivatives of  $X$  and  $Y$ , respectively,  $X_0 \in \text{int } \mathcal{K}$ ,  $Y_0 \in \text{int } \mathcal{K}$ ,  $P(X_0) + Q(Y_0) - a = 0$  and

$$H_P(U) := \frac{1}{2} \left( PUP^{-1} + (PUP^{-1})^T \right).$$

While the above ODEs define a feasible off-central path, infeasible off central paths have been also analyzed in [109].

At present, the following condition is indispensable to discuss superlinear or quadratic convergence of the infeasible interior point algorithms or analyticity of off-central paths.

**Condition 12.1 (Strictly complementarity solution)** *There exists a strictly complementary solution  $(X_*, Y_*)$  which is a solution of the SDCP satisfying*

$$X_* + Y_* \in \text{int } \mathcal{K} = \mathcal{S}_{++}^n := \{X \in \mathcal{S}^n \mid X > O\}$$

where  $X > O$  denotes that  $X$  is a positive definite matrix.

## 12.4 Merit or Smoothing Function Methods for the SCCP

Another general class of algorithms is so called merit or smoothing function methods which have been proposed especially for solving nonlinear complementarity problems. Let us consider the monotone vertical SCCP (12.5) satisfying (12.7). Here we introduce some important notions in the study of the merit function methods for the vertical SCCP.

**Definition 12.1 (Merit function for the SCCP).** A function  $f : \mathcal{V} \rightarrow [0, \infty)$  is said to be a merit function on  $\mathcal{U}$  for the SCCP if it satisfies

$$x \text{ is a solution of the SCCP} \iff f(x) = 0$$

on a set  $\mathcal{U}$  (typically  $\mathcal{U} = \mathcal{V}$  or  $\mathcal{U} = G^{-1}(\mathcal{K})$ ).

By using the merit function, we can reformulate the SCCP as the following minimization problem

$$\inf\{f(x) \mid x \in \mathcal{U}\}.$$

The merit function methods are to apply a feasible descent method to solve the above minimization problem.

A desirable property for a merit function is that the level sets are bounded. Another desirable property is that it gives an error bound defined as follows.

**Definition 12.2 (Error bound for the SCCP).** Let  $S$  be the solution set of the SCCP and  $\text{dist}(x, S)$  denote the distance between  $x \in \mathcal{V}$  and the set  $S$ . A function  $f : \mathcal{V} \rightarrow \mathfrak{R}$  is said to be a local error bound for the SCCP if there exist three constants  $\tau > 0$ ,  $\eta \in (0, 1]$  and  $\epsilon > 0$  such that

$$\text{dist}(x, S) \leq \tau f(x)^\eta \quad (12.30)$$

holds for any  $x \in S$  with  $f(x) \leq \epsilon$ .

The merit function is closely related to the following C-function (see, e.g., [25] in the case of the NCP).

**Definition 12.3 (C-function for the SCCP).** The function  $\Phi : \mathcal{V} \times \mathcal{V} \rightarrow \mathcal{V}$  satisfying

$$\langle x, y \rangle = 0, \quad x \in \mathcal{K}, \quad y \in \mathcal{K} \iff \Phi(x, y) = 0$$

is said to be a C-function for the SCCP.

It is clear that if  $\Phi : \mathcal{V} \times \mathcal{V} \rightarrow [0, \infty)$  is a C-function for the SCCP, then

$$f(x) := \|\Phi(F(x), G(x))\|^2$$

is a merit function for the vertical SCCP (12.5). For a C-function  $\Phi$ , the equation  $\Phi(x, y) = 0$  is called the *reformation equation* of the complementarity condition of the SCCP.

The C-function  $\Phi$  is typically nonsmooth but semismooth. In what follows, we introduce the notion of semismoothness of a function according to a fundamental paper by Sun and Sun [114] where some special functions (Löwner operator described in Sect. 12.4.1) over symmetric cones have been studied intensively.

Let  $\mathcal{U}$  and  $\mathcal{V}$  be two finite dimensional inner product space over the field  $\mathfrak{R}$ . Let  $\mathcal{W} \subseteq \mathcal{U}$  be an open subset of  $\mathcal{U}$  and  $\Phi : \mathcal{W} \rightarrow \mathcal{V}$  be a locally Lipschitz continuous function on the open set  $\mathcal{W}$ . By Rademachers theorem,  $\Phi$  is almost everywhere (in the sense of Lebesgue measure) differentiable (in the sense of Fréchet) in  $\mathcal{W}$ . Let  $\mathcal{D}_\Phi$  be the set of points in  $\mathcal{W}$  where  $\Phi$  is differentiable. Let  $\Phi'(x)$ , which is a linear mapping from  $\mathcal{U}$  to  $\mathcal{V}$ , denote the derivative of  $\Phi$  at  $x \in \mathcal{D}_\Phi$ . Then, the *B-subdifferential* of  $\Phi$  at  $x \in \mathcal{W}$ , denoted by  $\partial_B \Phi(x)$ , is the set of  $\{\lim_{k \rightarrow \infty} \Phi'(x_k)\}$ , where  $\{x_k\} \in \mathcal{D}_\Phi$  is a sequence converging to  $x$ . *Clarke's generalized Jacobian* of  $\Phi$  at  $x$  is the convex hull of  $\partial_B \Phi(x)$  (see [23]), i.e.,  $\partial \Phi(x) := \text{conv}\{\partial_B \Phi(x)\}$ . If  $S$  is any set of Lebesgue measure zero in  $\mathcal{U}$ , then

$$\partial\Phi(x) = \text{conv} \left\{ \lim_{k \rightarrow \infty} \Phi'(x_k) \mid x_k \rightarrow x, x_k \in \mathcal{D}_\Phi, x_k \notin S \right\} \quad (12.31)$$

(see Theorem 2.5.1 in [23]). Semismoothness was originally introduced by Mifflin [79] for functionals, and was used to analyze the convergence of bundle type methods for nondifferentiable optimization problems [80].

**Definition 12.4 (Semismoothness).** Let  $\Phi : \mathcal{W} \subseteq \mathcal{U} \rightarrow \mathcal{V}$  be a locally Lipschitz continuous function on the open set  $\mathcal{W}$ . Let us consider the following conditions for a fixed  $x \in \mathcal{W}$ .

- (i)  $\Phi$  is directionally differentiable at  $x$ .
- (ii) For any  $y \rightarrow x$  and  $V_y \in \partial\Phi(y)$  which depends on  $y$ ,

$$\Phi(y) - \Phi(x) - V_y(y - x) = o(\|y - x\|).$$

- (iii) There exists  $\rho > 0$  such that for any  $y \rightarrow x$  and  $V_y \in \partial\Phi(y)$  which depends on  $y$ ,

$$\Phi(y) - \Phi(x) - V_y(y - x) = O(\|y - x\|^{1+\rho}).$$

If  $\Phi$  satisfies (i) and (ii) at  $x$  then  $\Phi$  is said to be *semismooth* at  $x$ .

If  $\Phi$  is semismooth and satisfies (iii) at  $x$  then  $\Phi$  is said to be  $\rho$ -order *semismooth* at  $x$ .

If  $\Phi$  is 1-order semismooth at  $x$  then  $\Phi$  is said to be *strongly semismooth* at  $x$ .

If  $\Phi$  is semismooth (respectively,  $\rho$ -order semismooth) at any point of  $Q \subseteq \mathcal{W}$  then  $\Phi$  is said to be semismooth (respectively,  $\rho$ -order semismooth) on the set  $Q$ .

Many approaches have been proposed for approximating the C-function  $\Phi$  by a so-called *smoothing function*  $\Phi_\mu$  introducing a *smoothing parameter*  $\mu > 0$  in order to apply Newton-type methods to the function. By using a C-function or a smoothing function, the vertical SCCP (12.5) can be reformulated or approximated as the problem to find a solution of the system

$$\Phi(F(x), G(x)) = 0 \quad \text{or} \quad \Phi_\mu(F(x), G(x)) = 0.$$

While the smoothness of the function suggests the Newton method to solve the system, the semismoothness of the function allows us to adopt a Newton-type method, so-called *semismooth Newton method*, which is a significant property of the semismooth functions (see Chap. 7 of [25] for the case  $\mathcal{K} = \mathbb{R}_+^n$ ). We call such methods *smoothing function methods*. The smoothing parameter  $\mu > 0$  is often dealt as a variable in smoothing function methods.

The merit or smoothing function methods for the SCCP were first proposed for the SDCP as a special case, followed by the SOCCP and the SCCP. We will give a brief survey on the studies chronologically according to research progress. Since the SCCP includes the SDCP and the SOCCP, the results obtained for the SCCP hold for the SDCP and the SOCCP as well.

### 12.4.1 Merit or Smoothing Function Methods for the SDCP

A pioneer work on the study of merit function method for solving the SDCP has been done by Tseng [119]. For a while, we consider that  $\mathcal{S}^n$  is the set of all  $n \times n$  symmetric matrices and  $\mathcal{K}$  is the positive semidefinite cone. The inner product and the norm are given by

$$\langle X, Y \rangle := \text{Tr}(X^T Y), \quad \|X\| := \sqrt{\langle X, X \rangle}.$$

It has been shown in [119] that each of the following functions is a merit function for the vertical SDCP (12.5).

**Natural residual function** (Proposition 2.1 of [119]) Let the function  $[.]_+$  denote the orthogonal projection onto  $\mathcal{S}_+^n$ , i.e.,

$$[X]_+ = \arg \min_{W \in \mathcal{S}_+^n} \|X - W\| \quad (12.32)$$

and define the natural residual function on  $\mathcal{S}^n$  as

$$\Phi^{\text{NR}}(A, B) := B - [B - A]_+ \quad (12.33)$$

Then the function

$$f(X) := \|\Phi^{\text{NR}}(F(X), G(X))\|^2 \quad (12.34)$$

is a merit function on  $\mathcal{S}^n$  for the vertical SDCP (12.5). The natural residual function has been studied in [24, 72, 76, 94] for the NCP.

**Fischer–Burmeister function** (Proposition 6.1 of [119]) Let  $\phi^{\text{FB}} : \mathcal{S}^n \times \mathcal{S}^n \rightarrow \mathcal{S}^n$  be the Fischer–Burmeister function on  $\mathcal{S}^n$  given by

$$\Phi^{\text{FB}}(A, B) := (A + B) - (A^2 + B^2)^{1/2}. \quad (12.35)$$

Then the function

$$f(X) := \frac{1}{2} \|\Phi^{\text{FB}}(F(X), G(X))\|^2 \quad (12.36)$$

is a merit function on  $\mathcal{S}^n$  for the vertical SDCP (12.5). The Fischer–Burmeister function has been proposed by Fischer [29–31] for the NCP.

The fact that the functions (12.35) and (12.37) are merit functions of the vertical SDCP (12.5) implies that  $\Phi^{\text{NR}}(x, y)$  and  $\Phi^{\text{FB}}(x, y)$  are C-function for the vertical SDCP (12.5).

For each of the above functions  $f$ , Tseng [119] derived conditions on  $F$  and  $G$  for  $f$  to be convex and/or differentiable, and for the stationary point of  $f$  to be a solution of the SDCP.

On the other hand, Chen and Tseng [17] introduced a wide class of functions so called Chen and Mangasarian smoothing functions [9, 10, 120].

**Chen and Mangasarian smoothing functions** Let  $g : \mathbb{R} \rightarrow \mathbb{R}$  be a univariate function satisfying

$$\lim_{\tau \rightarrow -\infty} g(\tau) = 0, \quad \lim_{\tau \rightarrow \infty} g(\tau) - \tau = 0, \quad 0 < g'(\tau) < 1. \quad (12.37)$$

For each symmetric matrix  $X$  having the spectral decomposition

$$X = \sum_{i=1}^r \lambda_i(X) u_i u_i^T \quad (12.38)$$

where  $\lambda_i(X)$  and  $u_i$  are the eigenvalues and the corresponding eigenvectors of  $X$ , define the function  $g_{\mathcal{S}^n} : \mathcal{S}^n \rightarrow \mathcal{S}^n$  as

$$g_{\mathcal{S}^n}(X) := \sum_{i=1}^r g(\lambda_i(X)) u_i u_i^T. \quad (12.39)$$

Then the function

$$\Phi_\mu(X, Y) := X - \mu g_{\mathcal{S}^n}((X - Y)/\mu) \quad (12.40)$$

with a positive parameter  $\mu > 0$  is a function in the class of Chen and Mangasarian smoothing functions.

The function  $\Phi_\mu$  can be regarded as a smoothed approximation to the natural residual function  $\Phi^{\text{NR}}$  by a smoothing parameter  $\mu > 0$ . The function  $\Phi_\mu$  is continuously differentiable having the property that

$$\Phi_\mu(A, B) \rightarrow 0 \text{ and } (A, B, \mu) \rightarrow (X, Y, 0) \implies X \in \mathcal{S}_+^n, Y \in \mathcal{S}_+^n, \langle X, Y \rangle = 0.$$

If we choose

$$g(\tau) := ((\tau^2 + 4)^{1/2} + \tau)/2 \quad (12.41)$$

then the function  $\Phi_\mu$  is an extension of Chen and Harker [7, 8], Kanzow [49, 50] and Smale [110] (CHKS) smoothing function for the NCP and if we choose

$$g(\tau) := \ln(e^\tau + 1)$$

then the function  $\Phi_\mu$  is an extension of Chen and Mangasarian [9, 10] smoothing function for the NCP. In [17], the authors provided a smoothing continuation method using the function  $\Phi_\mu$  and studied various issues on existence of Newton directions, boundedness of iterates, global convergence, and local superlinear convergence. They also reported a preliminary numerical experience on semidefinite linear programs.

Note that the function  $g_{\mathcal{S}^n} : \mathcal{S}^n \rightarrow \mathcal{S}^n$  in (12.39) is constructed from a univariate function  $g : \mathbb{R} \rightarrow \mathbb{R}$ . Such a function has been introduced by Löwner [70] in 1934,

and we call the function  $g_{\mathcal{S}^n}$  the Löwner operator. See also [57, 114]. It is known that the orthogonal projection (12.32) onto  $\mathcal{S}_+^n$  is given by

$$[X]_+ = \sum_{i=1}^n [\lambda_i(X)]_+ u_i u_i^T \quad (12.42)$$

where  $X$  has the spectral decomposition (12.38) and  $[\cdot]_+ : \mathfrak{R} \rightarrow \mathfrak{R}$  is the plus function  $[t]_+ := \max\{0, t\}$  [119]. This fact implies that  $[\cdot]_+$  is also a Löwner operator.

Several papers have been focused on the properties of Löwner operators. Sun and Sun [111] showed that the projection function  $[\cdot]_+ : \mathcal{S}^n \rightarrow \mathcal{S}^n$  and the function  $\Phi^{\text{NR}}$  in (12.34) are strongly semismooth on  $\mathcal{S}^n$  (see Definition 12.4). Chen and Tseng [18] extended the results and investigated that the Löwner operator  $g_{\mathcal{S}^n}$  inherits from  $g$  the properties of continuity, (local) Lipschitz continuity, directional differentiability, Fréchet differentiability, continuous differentiability, as well as  $\rho$ -order semismoothness in the case of the SDCP. These results have been extended to the function  $g_V$  over the Euclidean Jordan algebra by Sun and Sun [114], which we will describe in a little more detail in Sect. 12.4.3.

For the Fischer–Burmeister function  $\Phi^{\text{FB}} : \mathcal{S}^n \times \mathcal{S}^n \rightarrow \mathcal{S}^n$  in (12.36), Sun and Sun [113] showed that  $\Phi^{\text{FB}}$  is globally Lipschitz continuous, directionally differentiable, and strongly semismooth.

See [119] and [122] for other merit functions proposed for the SDCP. Further developments also can be seen in [6, 46, 103, 127], etc. Meanwhile, besides the paper [17], there are many papers on smoothing continuation methods for the SDCP, e.g., see [18, 45, 51, 112, 113].

### 12.4.2 Merit or Smoothing Function Methods for the SOCCP

A first study on smoothing function methods for solving the SOCCP has been provided by Fukushima et al. [32]. For the implicit SOCCP (12.3), the authors introduced the class of Chen and Mangasarian smoothing functions, studied the Lipschitzian and differential properties of the functions, and derived computable formulas for the functions and their Jacobians. They also showed the existence and uniqueness of the Newton direction when the mapping  $F$  is monotone.

In [19], the authors introduced a C-function for the standard SOCCP (12.2) and showed that the squared C-function is strongly semismooth. They also reported numerical results of the squared smoothing Newton algorithms.

Chen et al. [11] showed that the Löwner operator  $g_V$  for the standard SCCP (12.2) inherits from  $g$  the properties of continuity, (local) Lipschitz continuity, directional differentiability, Fréchet differentiability, continuous differentiability, as well as  $\rho$ -order semismoothness.

The SOCCP has a simple structure, i.e., the rank  $r$  of the second order cone is always  $r = 2$ . This may be a reason why the SOCCP has many applications, e.g, the *three dimensional quasi-static frictional contact* [48] and the *robust Nash equilibria* [43, 87].

See [5, 12–16, 26, 39, 88, 91, 93, 96, 115, 126] for many other merit function methods for the SOCCP, [44, 113] for smoothing continuation methods for the SOCCP and [59] for the solution set structure of the monotone SOCCP.

### 12.4.3 Merit or Smoothing Function Methods for the SCCP

The merit or smoothing function methods for the SCCP have become a very active research area in recent few years. Most of the results for the SDCP and the SOCCP have been extended to the SCCP.

A first C-function for the SCCP has been proposed by Gowda [35] where some  $\mathbf{P}$ -properties for the SCCP have been discussed (see Sect. 12.5). The following proposition ensures that the natural residual function  $\Phi^{\text{NR}}(x, y) := x - [x - y]_+$  and the Fischer–Burmeister function  $\Phi^{\text{FB}}(x, y) := x + y - \sqrt{x^2 + y^2}$  are C-functions for the SCCP (see Proposition 6 of [35]).

**Proposition 12.2 (C-functions for the SCCP).** *For  $(x, y) \in \mathcal{V}$ , the following conditions are equivalent.*

- (a)  $x - [x - y]_+ = 0$ .
- (b)  $(x, y) \in \mathcal{K} \times \mathcal{K}$  and  $\langle x, y \rangle = 0$ .
- (c)  $(x, y) \in \mathcal{K} \times \mathcal{K}$  and  $x \circ y = 0$ .
- (d)  $x + y - \sqrt{x^2 + y^2} = 0$ .
- (e)  $x + y \in \mathcal{K}$  and  $x \circ y = 0$ .

Here  $[x]_+$  denotes the orthogonal projection onto  $\mathcal{K}$ , i.e.,  $[x]_+ = \arg \min_{w \in \mathcal{K}} \|x - w\|$ .

Suppose that  $x \in \mathcal{V}$  has the spectral decomposition  $x = \sum_{i=1}^r \lambda_i(x) c_i$  (see Theorem 12.2). For a given univariate function  $\phi : \mathfrak{R} \rightarrow \mathfrak{R}$ , we define the vector-valued function  $\phi_{\mathcal{V}} : \mathcal{V} \rightarrow \mathcal{V}$  associated with the Euclidean Jordan algebra  $(\mathcal{V}, \circ, \langle \cdot, \cdot \rangle)$  by

$$\phi_{\mathcal{V}}(x) := \sum_{i=1}^r \phi(\lambda_i(x)) c_i. \quad (12.43)$$

The function  $\phi_{\mathcal{V}}$  is the (extended) Löwner operator for which Korányi [64] extended Löwner's results on the monotonicity of  $\phi_{\mathcal{S}^n}$  to  $\phi_{\mathcal{V}}$ . For an example, the projection function  $[\cdot]_+ : \mathcal{V} \rightarrow \mathcal{V}$  can be represented by

$$[x]_+ = \sum_{i=1}^r [\lambda_i(x)]_+ c_i$$

where  $[\cdot]_+ : \mathfrak{R} \rightarrow \mathfrak{R}$  is the plus function  $[t]_+ := \max\{0, t\}$  (see [35]).

The fundamental paper of Sun and Sun [114] gives basic properties of Löwner operators over  $(\mathcal{V}, \circ, \langle \cdot, \cdot \rangle)$  on differentiability and semismoothness. Here we refer the following important results (see Theorem 3.2, Propositions 3.3, 3.4 and Theorem 3.3 of [114]).

**Theorem 12.8 (Differentiability of  $\phi_{\mathcal{V}}$ ).** *Let  $x = \sum_{i=1}^r \lambda_i(x) c_i$ . The function  $\phi_{\mathcal{V}} : \mathcal{V} \rightarrow \mathcal{V}$  is differentiable (respectively, continuously differentiable) at  $x$  if and only if for each  $i \in \{1, 2, \dots, r\}$ ,  $\phi : \mathbb{R} \rightarrow \mathbb{R}$  is differentiable (continuously differentiable) at  $\lambda_i(x)$ .*

**Theorem 12.9 (Semismoothness of  $\phi_{\mathcal{V}}$ ).**

- (i) *The projection  $[\cdot]_+ : \mathcal{V} \rightarrow \mathcal{V}$  is strongly semismooth on  $\mathcal{V}$ .*
- (ii) *For  $\epsilon \in \mathbb{R}$ , define  $\phi^\mu : \mathbb{R} \rightarrow \mathbb{R}$  by*

$$\phi^\mu(t) := \sqrt{t^2 + \mu^2}.$$

*The corresponding Löwner operator is given by*

$$\phi_{\mathcal{V}}^\mu(x) := \sum_{i=1}^r \sqrt{\lambda_i(x)^2 + \mu^2} c_i = \sqrt{x^2 + \mu^2 e}$$

*which is a smoothed approximation to the absolute value function  $|x| := \sqrt{x^2}$  (see also (12.41) and (12.42)). Let  $\psi(\mu, x) := \phi_{\mathcal{V}}^\mu(x)$ . Then  $\psi : \mathbb{R} \times \mathcal{V} \rightarrow \mathcal{V}$  is continuously differentiable at  $(\mu, x)$  if  $\mu \neq 0$  and is strongly semismooth at  $(0, x)$ ,  $x \in \mathcal{V}$ .*

- (iii) *Let  $\rho \in (0, 1]$  be a constant and  $x = \sum_{i=1}^r \lambda_i(x) c_i$ . The function  $\phi_{\mathcal{V}} : \mathcal{V} \rightarrow \mathcal{V}$  is semismooth (respectively,  $\rho$ -order semismooth) if and only if for each  $i \in \{1, 2, \dots, r\}$ ,  $\phi : \mathbb{R} \rightarrow \mathbb{R}$  is semismooth ( $\rho$ -order semismooth) at  $\lambda_i(x)$ .*

Kong et al. [57] investigated the connection between the monotonicity of the function  $\phi$  and its corresponding Löwner operator. For a given real interval  $(a, b)$  with  $a < b$  ( $a, b \in \mathbb{R} \cup \{-\infty\} \cup \{\infty\}$ ), denote by  $\mathcal{V}(a, b)$  the set of all  $x \in \mathcal{V}$  such that  $x - ae, be - x \in \text{int } K$ . They showed the following properties (see Theorem 3 of [57]).

**Theorem 12.10 (Convexity of  $\phi_{\mathcal{V}}$ ).** *Let  $g$  be a locally Lipschitz function from  $(a, b)$  into  $\mathbb{R}$ .  $\phi_{\mathcal{V}}$  is monotone (respectively, strictly monotone, strongly monotone) on  $\mathcal{V}(a, b)$  if and only if  $\phi$  is monotone (respectively, strictly monotone, strongly monotone) on  $(a, b)$ .*

Many studies have been conducted on merit or smoothing function methods for the SCCP and some of them are still continuing even today. The subjects dealt in these studies are the natural residual function [62], the Fischer–Burmeister (smoothing) function [4, 58, 69], Chen–Mangasarian smoothing functions [22, 47, 60], other merit functions [41, 56, 61, 65–68, 89, 92], and smoothing continuation methods [21, 22, 47, 60, 86, 123], etc.

## 12.5 Properties of the SCCP

In the discussions so far, we assume that the SCCP is monotone. In this chapter, we observe other various properties of the SCCP. We address the following two SCCPs,  $\text{LCP}(L, q)$  and  $\text{NLCP}(\psi, q)$ , which are alternate representations of the standard linear or nonlinear SCCP (12.2).

$$\left| \begin{array}{l} \text{LCP}(L, q) \text{ Find } x \in \mathcal{K} \\ \text{s.t. } x \in \mathcal{K}, Lx + q \in \mathcal{K}, \langle x, Lx + q \rangle = 0 \end{array} \right. \quad (12.44)$$

where  $q \in \mathcal{V}$  and  $L : \mathcal{V} \rightarrow \mathcal{V}$  is a linear operator.

$$\left| \begin{array}{l} \text{NLCP}(\psi, q) \text{ Find } x \in \mathcal{K} \\ \text{s.t. } x \in \mathcal{K}, \psi(x) + q \in \mathcal{K}, \langle x, \psi(x) + q \rangle = 0 \end{array} \right. \quad (12.45)$$

where  $q \in \mathcal{V}$  and  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  is a (nonlinear) continuous function.

The linear operator  $L$  and  $q \in V$  determine the property of  $\text{LCP}(L, q)$ . Gowda et al. [35] focused their attention on the **P**-property of the linear operator  $L$ . If we restrict  $\mathcal{V} = \mathbb{R}^n$  and  $\mathcal{K} = \mathbb{R}_+^n$  then the **P**-property of the linear operator  $L$  can be characterized as follows (see [3, 24, 25], etc.).

**Proposition 12.3 (P-property of the LCP over  $\mathbb{R}_+^n$ ).** *Let  $\mathcal{V} = \mathbb{R}^n$  and  $\mathcal{K} = \mathbb{R}_+^n$ . Then the following properties of the linear operator  $L : \mathbb{R}^n \rightarrow \mathbb{R}^n$  are equivalent.*

(i) All principal minors of  $L$  are positive.

(ii)

$$x \in \mathbb{R}^n, x \circ Lx \leq 0 \implies x = 0$$

where  $a \circ b$  denotes the componentwise product for  $a, b \in \mathbb{R}^n$ .

(iii)

$$x \wedge Lx \leq 0 \leq x \vee Lx \implies x = 0$$

where  $a \wedge b$  and  $a \vee b$  denote the componentwise minimum and maximum of  $a, b \in \mathbb{R}^n$ , respectively.

(iv) For all  $q \in \mathbb{R}^n$ , there exists a unique  $x \in \mathbb{R}^n$  such that

$$x \geq 0, Lx + q \geq 0, \langle x, Lx + q \rangle = 0.$$

(v) The function  $F(x) := L[x]_+ + x - [x]_+$  is invertible in a neighborhood of zero.

(vi) The function  $F(x) := L[x]_+ + x - [x]_+$  is invertible in a neighborhood of zero with Lipschitzian inverse.

Note that the function  $F(x) := L[x]_+ + x - [x]_+$  is called the *normal map* of the SCCP and known as a C-function for the SCCP (see [21] and also see Sect. 1.5 of [25] in the case of the NCP). In contrast, the above properties are different for the SCCP in general. How should we define the **P**-property for the SCCP? Gowda et al.[35] introduced the following properties (i)-(viii) of the linear operator  $L : \mathcal{V} \rightarrow \mathcal{V}$  in the setting of a Euclidean Jordan algebra.

**Definition 12.5 (P-properties of  $L: V \rightarrow V$ ).** The linear operator  $L: V \rightarrow V$  is said to be (to have):

- (i) **Monotone** (respectively, **strictly monotone**) if  $\langle Lx, x \rangle \geq 0$  (respectively,  $\langle Lx, x \rangle > 0$ ) for all  $0 \neq x \in V$ ,
- (ii) The **order P**-property if

$$x \sqcap Lx \in -K, x \sqcup Lx \in K \implies x = 0,$$

where  $x \sqcap y := x - [x - y]_+$  and  $x \sqcup y := y + [x - y]_+$ ,

- (iii) The **Jordan P**-property if

$$x \circ Lx \in -K \implies x = 0,$$

- (iv) The **P**-property if

$$\left. \begin{array}{l} x \text{ and } Lx \text{ operator commute} \\ x \circ Lx \in -K \end{array} \right\} \implies x = 0,$$

- (v) The **Globally Uniquely Solvable (GUS)** if any  $q \in V$ ,  $LCP(L, q)$  has a unique solution.
- (vi) The **Cross Commutative property (Cross Commutative)** if for any  $q \in V$  and for any two solution  $x_1$  and  $x_2$  of  $LCP(L, q)$ ,  $x_1$  operator commutes with  $y_2 = Lx_2 + q$  and  $x_2$  operator commutes with  $y_1 = Lx_1 + q$ .
- (vii) The **Lipschitzian GUS**-property if the normal map  $F(x) := L[x]_+ + x - [x]_+$  is a homeomorphism of  $V$  and the inverse of  $F$  is Lipschitzian.
- (viii) The **positive principal minor (positive PM)** if all principal minors of  $L$  are positive.

Gowda et al. [35] showed the following implications (see Theorems 11, 12, 14, 17 and 23 and Examples 1.3 and 2.3 of [35]).

**Proposition 12.4 (Implications among the properties on  $L$ ).**

- (i) **Strictly monotonicity**  $\implies$  **Order P**  $\implies$  **Jordan P**.
- (ii) *If  $L$  has **P**-property then the every real eigenvalue of  $L$  is positive.*
- (iii) *If  $L$  has **P**-property then for any  $q \in V$ ,  $LCP(L, q)$  has a nonempty compact solution set.*
- (iv) **GUS**  $=$  **P** + **Cross Commutative**.
- (v) **Strictly monotone**  $\implies$  **Lipschitzian GUS**  $\implies$  **positive PM**.
- (vi) **GUS**  $\not\implies$  **Lipschitzian GUS**.
- (vii) **Order P**  $\not\implies$  **strong monotonicity**.
- (viii) *If  $K$  is polyhedral then, **order P** = **Jordan P** = **P** = **GUS** = **positive PM**.*

Moreover, Tao and Gowda [116] extended the above results to  $NLCP(\psi, q)$  introducing the following properties.

**Definition 12.6 (P-properties of  $\psi : V \rightarrow V$ ).** The continuous function  $\psi : V \rightarrow V$  is said to be (to have):

- (i) **Monotone** if for all  $x, y \in V$ ,  $\langle x - y, \psi(x) - \psi(y) \rangle \geq 0$ .
- (ii) **Strictly monotone** if for all  $x \neq y \in V$ ,  $\langle x - y, \psi(x) - \psi(y) \rangle > 0$ .
- (iii) **Strongly monotone** if there is an  $\alpha > 0$  such that for all  $x, y \in V$ ,

$$\langle x - y, \psi(x) - \psi(y) \rangle \geq \alpha \|x - y\|^2,$$

- (iv) The **order P**-property if

$$(x - y) \sqcap (\psi(x) - \psi(y)) \in -K, (x - y) \sqcup (\psi(x) - \psi(y)) \in K \implies x = y,$$

- (v) The **Jordan P**-property if

$$(x - y) \circ (\psi(x) - \psi(y)) \in -K \implies x = y,$$

- (vi) The **P**-property if

$$\left. \begin{array}{l} x - y \text{ and } \psi(x) - \psi(y) \text{ operator commute} \\ (x - y) \circ (\psi(x) - \psi(y)) \in -K \end{array} \right\} \implies x = y,$$

- (vii) The **uniform Jordan P**-property if there is an  $\alpha > 0$  such that for all  $x, y \in V$ ,

$$\lambda_{\max}((x - y) \circ (\psi(x) - \psi(y))) \geq \alpha \|x - y\|^2,$$

- (viii) The **uniform P**-property if there is an  $\alpha > 0$  such that for all  $x, y \in V$  with  $x - y$  operator commuting with  $\psi(x) - \psi(y)$ ,

$$\lambda_{\max}((x - y) \circ (\psi(x) - \psi(y))) \geq \alpha \|x - y\|^2,$$

- (ix) The **P<sub>0</sub>**-property if  $\psi(x) + \epsilon x$  has the **P**-property for all  $\epsilon > 0$ ,

- (x) The **R<sub>0</sub>**-property if for any sequence  $\{x_k\}$  in  $V$  with

$$\|x_k\| \rightarrow \infty, \liminf_{k \rightarrow \infty} \frac{\lambda_{\min}(x_k)}{\|x_k\|} \geq 0, \liminf_{k \rightarrow \infty} \frac{\lambda_{\min}(\psi(x_k))}{\|x_k\|} \geq 0,$$

we have

$$\liminf_{k \rightarrow \infty} \frac{\langle x_k, \psi(x_k) \rangle}{\|x_k\|^2} \geq 0.$$

Here  $\lambda_{\max}(x)$  and  $\lambda_{\min}(x)$  denote the maximum and the minimum eigenvalues of  $x$ , respectively.

Tao and Gowda [116] showed the following implications (see Proposition 3.1, Theorem 3.1, Propositions 3.2 and 3.3, Corollary 3.1 and Theorem 4.1 of [116]).

**Proposition 12.5 (Implications among the properties on  $\psi$ ).**

- (i) **Strong monotonicity**  $\Rightarrow$  **strict monotonicity**  $\Rightarrow$  **order P**  $\Rightarrow$  **Jordan P**  $\Rightarrow$  **P**  $\Rightarrow$  **P<sub>0</sub>**.
- (ii) **Strong monotonicity**  $\Rightarrow$  **uniform Jordan P**  $\Rightarrow$  **uniform P**.
- (iii) **Monotonicity**  $\Rightarrow$  **P<sub>0</sub>**.
- (iv) If  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  has the **P<sub>0</sub>-property** and for any  $\delta > 0$  in  $\mathbb{R}$  and the set

$$\{x \in \mathcal{K} \mid x \text{ solves } NLCP(\psi, q), \|q\| \leq \delta\} \quad (12.46)$$

is bounded, then for any  $q \in \mathcal{V}$ ,  $NLCP(\psi, q)$  has a nonempty bounded solution set.

- (v) If  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  has the **R<sub>0</sub>-property**, then for any  $\delta > 0$  in  $\mathbb{R}$ , the set (12.45) is bounded,
- (vi) If  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  has the **P<sub>0</sub>** and **R<sub>0</sub>** property, then for all  $q \in \mathcal{V}$ , the solution set of  $NLCP(\psi, q)$  is nonempty and bounded. Moreover, there exists an  $x \in \text{int } \mathcal{K}$  such that  $\psi(x) + q \in \text{int } \mathcal{K}$ .
- (vii) If  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  has the **GUS**-property, then for any primitive idempotent  $c \in \mathcal{V}$ ,  $\langle \psi(c) - \psi(0), c \rangle \geq 0$ .

See also [33, 34] for the case of the SDCP.

Kong et al. [63] extended the **P**-property to the linear complementarity problem over *homogeneous cones*. Note that homogeneous cones properly include symmetric cones and if a homogeneous cone is self-dual then it is a symmetric cone.

Tao and Gowda [116] also introduced an approach to construct a relaxation transformation based on the Peirce decomposition of a given  $x \in \mathcal{V}$  (see Theorem 12.3). Suppose we are given a Jordan frame  $\{c_1, c_2, \dots, c_r\}$  in  $\mathcal{V}$  and a continuous function  $\phi : \mathbb{R}^r \rightarrow \mathbb{R}^r$ ,  $\phi(u) := [\phi_1(u), \phi_2(u), \dots, \phi_r(u)]$  for  $u \in \mathbb{R}^r$ . We can write any  $x \in \mathcal{V}$  as the Peirce decomposition

$$x = \sum_{i=1}^r x_i c_i + \sum_{i < j} x_{ij} .$$

where  $x_i \in \mathbb{R}$  and  $x_{ij} \in \mathcal{V}_{ij} := \{x \in \mathcal{V} \mid x \circ c_i = \frac{1}{2}x = x \circ c_j\}$ . Then the *relaxation transformation*  $R_\phi : \mathcal{V} \rightarrow \mathcal{V}$  is given by

$$R_\phi(x) := \sum_{i=1}^r \phi_i([x_1, x_2, \dots, x_r]) c_i + \sum_{i < j} x_{ij} . \quad (12.47)$$

The following **P**-properties of the relaxation transformation  $R_\phi$  have been shown (see Propositions 5.1 and 5.2 of [116]).

**Proposition 12.6 (P-properties of the relaxation transformation  $R_\phi$ ).** *The following statements are equivalent.*

- (a)  $\phi$  is a **P**-function.
- (b)  $R_\phi$  has the **order P**-property.
- (c)  $R_\phi$  has the **Jordan P**-property.
- (d)  $R_\phi$  has the **P**-property.

**Proposition 12.7 (NLCP( $R_\phi, q$ ) having a nonempty and bounded solution set).**

If  $\phi : \mathbb{R}^r \rightarrow \mathbb{R}^r$  has the **P<sub>0</sub>**- and **R<sub>0</sub>**-property then for the relaxation transformation  $R_\phi$ , the set (12.47) is bounded for any  $\delta > 0$ . Hence, by (iii) of Proposition 12.5, NLCP( $R_\phi, q$ ) has a nonempty and bounded solution set.

Proposition 12.7 implies a significance of the relaxation transformation  $R_\phi$ : Only by investigating the vector valued function  $\phi$ , we can find whether the solution set of the associated SCCP is nonempty and bounded.

Lu et al. [71] stated the differences between the Löwner operator (12.44) and the relaxation transformation (12.47) as follows.

Both the relaxation transformation and the Löwner operator are the generalization of functions defined in the Euclidean vector space. However, they are more difficult to be discussed than those in the Euclidean vector space. The main difficulty from the Löwner operator is that different elements are given by the different Jordan frames; while the main difficulty from the relaxation transformation is that every transformation  $R_\phi$  has an additional item  $\sum_{i < j} x_{ij}$  though a common Jordan frame is used. In addition, there are many differences between the relaxation transformation and the Löwner operator. For example, the Löwner operator  $\phi_V(x)$  is defined based on the spectral decomposition of  $x$ , and hence,  $\phi_V(x)$  operator commutes with  $x$ ; while the relaxation transformation  $R_\phi$  is defined based on the Peirce decomposition of  $x$ , and it is easy to see that  $R_\phi$  operator does not commute with  $x$  because of the existence of an additional item  $\sum_{i < j} x_{ij}$ .

They investigated some inter connections between  $\phi$  and  $R_\phi$  concerning continuity, differentiability, semismoothness, monotonicity and **P**-properties. They also provided a smoothing method for solving NLCP( $R_\phi, q$ ) based on the CHKS smoothing function and showed its global convergence. See [36–38, 117, 118] for more recent results on these functions.

Another property, sufficiency, of the linear operator  $L$  in LCP( $R_\phi, q$ ) has been observed in [100].

**Definition 12.7 (Sufficient properties of  $L : V \rightarrow V$ ).** The linear operator  $L : V \rightarrow V$  is said to have:

- (i) The **Jordan column-sufficient (Jordan CSU)** property if

$$x \circ Lx \in -K \implies x \circ Lx = 0,$$

- (ii) The **column-sufficient (CSU)** property if

$$\left. \begin{array}{l} x \text{ and } Lx \text{ operator commute} \\ x \circ Lx \in -K \end{array} \right\} \implies x \circ Lx = 0,$$

- (iii) The **Jordan row-sufficient (Jordan RSU)** property if the adjoint operator  $L^*$  of  $L$  has the Jordan column-sufficient property,
- (iii) The **row-sufficient (RSU)** property if the adjoint operator  $L^*$  of  $L$  has the column-sufficient property.

Then the following implications hold (see Remark (iii), Theorems 3.5, 4.3 of [100]).

**Theorem 12.11 (Implications among sufficient properties on  $L$ ).**

- (i)  $\mathbf{P} \implies \mathbf{CSU} \implies \mathbf{P}_0$ .
- (ii) For a linear operator  $L : \mathcal{V} \rightarrow \mathcal{V}$ , the following statements are equivalent
  - (a)  $L$  has the **CSU** and **Cross Commutative** properties.
  - (b) For any  $q \in \mathcal{V}$ , the solution set of  $LCP(L, q)$  is convex (possibly empty).
- (iii) For a linear operator  $L : \mathcal{V} \rightarrow \mathcal{V}$ , the following statements are equivalent
  - (a)  $L$  has the **RSU** property.
  - (b) For any  $q \in \mathcal{V}$ , if the KKT point  $(x, u)$  of the problem

$$\begin{aligned} & \min \frac{1}{2} \langle x, Lx + L^*x \rangle + \langle q, x \rangle \\ & s.t. x \in K, Lx + q \in K \end{aligned}$$

satisfies that  $(x - u)$  and  $L^*(x - u)$  operator commute, then  $x$  solves  $LCP(L, q)$ .

Suppose that  $\mathcal{V}$  is the Cartesian product space  $\mathcal{V} := \mathcal{V}_1 \times \mathcal{V}_2 \times \cdots \times \mathcal{V}_m$  with its symmetric cone  $\mathcal{K} := \mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_m$ . Then the following properties of the linear operator  $L$  in  $LCP(L, q)$  and the nonlinear function  $\psi$  in  $NLCP(\psi, q)$  have been introduced in [20, 61, 75, 90], etc.

**Definition 12.8 (Cartesian P properties of  $L$  and  $\psi$ ).** The linear operator  $L : \mathcal{V} \rightarrow \mathcal{V}$  is said to have:

- (i) The **Cartesian P**-property if for any nonzero  $x = (x_1, x_2, \dots, x_m) \in \mathcal{V}$  with  $x_i \in \mathcal{V}_i$ , there exists an index  $v \in \{1, 2, \dots, m\}$  such that  $\langle x_v, (Lx)_v \rangle > 0$ ,
- (ii) The **Cartesian  $P_0$** -property if for any nonzero  $x = (x_1, x_2, \dots, x_m) \in \mathcal{V}$  with  $x_i \in \mathcal{V}_i$ , there exists an index  $v \in \{1, 2, \dots, m\}$  such that  $x_v \neq 0$  and  $\langle x_v, (Lx)_v \rangle \geq 0$ ,
- (iii) The **Cartesian  $P_*(\kappa)$** -property if for any  $x = (x_1, x_2, \dots, x_m) \in \mathcal{V}$  with  $x_i \in \mathcal{V}_i$ ,

$$(1 + 4\kappa) \sum_{v \in I_+(x)} \langle x_v, (Lx)_v \rangle + \sum_{v \in I_-(x)} \langle x_v, (Lx)_v \rangle \geq 0$$

where

$$I_+(x) := \{v \mid \langle x_v, (Lx)_v \rangle > 0\}, \quad I_-(x) := \{v \mid \langle x_v, (Lx)_v \rangle < 0\}.$$

The function  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  is said to have:

- (iv) The **uniform Cartesian P**-property if there exists a constant  $\alpha$  such that for any  $x, y \in \mathcal{V}$ , there exists an index  $v \in \{1, 2, \dots, m\}$  such that  $\langle x_v - y_v, (\psi(x))_v - (\psi(y))_v \rangle \geq \alpha \|x - y\|^2$ ,
- (iv) The **Cartesian P**-property if for any  $x \neq y \in \mathcal{V}_i$ , there exists an index  $v \in \{1, 2, \dots, m\}$  such that  $\langle x_v - y_v, (\psi(x))_v - (\psi(y))_v \rangle > 0$ ,
- (v) The **Cartesian P<sub>0</sub>**-property if for any  $x \neq y \in \mathcal{V}_i$ , there exists an index  $v \in \{1, 2, \dots, m\}$  such that  $\langle x_v - y_v, (\psi(x))_v - (\psi(y))_v \rangle \geq 0$ ,
- (vi) The **Cartesian P<sub>\*</sub>(κ)**-property if for any  $x, y \in \mathcal{V}$ ,

$$(1 + 4\kappa) \sum_{v \in I_+(x)} \langle x_v - y_v, (\psi(x))_v - (\psi(y))_v \rangle \\ + \sum_{v \in I_-(x)} \langle x_v - y_v, (\psi(x))_v - (\psi(y))_v \rangle \geq 0$$

where

$$I_+(x) := \{v \mid \langle x_v - y_v, (\psi(x))_v - (\psi(y))_v \rangle > 0\},$$

$$I_-(x) := \{v \mid \langle x_v - y_v, (\psi(x))_v - (\psi(y))_v \rangle < 0\}.$$

The above properties have been motivated by a complete characterization of Euclidean Jordan algebras (see the Chap. 11 of this Handbook or Chap. V of [27]). For LCP( $L, q$ ) over the semidefinite cone  $\mathcal{K} = \mathcal{S}_+^n$ , it has been shown that the following implications

$$\text{Strong monotonicity} \implies \text{Cartesian P} \implies \begin{cases} \mathbf{P}, \\ \mathbf{GUS}. \end{cases}$$

hold for any  $L : \mathcal{V} \rightarrow \mathcal{V}$  [20]. In [90], a merit function based on the Fischer–Burmeister function has been proposed for solving NLCP( $\psi, q$ ) where  $\mathcal{K}$  is the second order cone  $\mathcal{L}^n$  and  $\psi$  has the **Cartesian P<sub>0</sub>**-property. In [75], a theoretical framework of path-following interior point algorithms has been established for solving LCP( $L, q$ ) in the setting of a Euclidean Jordan algebra, where  $L$  has the **Cartesian P<sub>\*</sub>(κ)**-property. The global convergence and the iteration complexities have been also discussed.

Furthermore, Chua et al. [22] (see also [21]) introduced the following property, *uniform nonsingularity*, of the nonlinear operator  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  in NLCP( $\psi, q$ ),

**Definition 12.9 (Uniform nonsingularity).** The function  $\psi : \mathcal{V} \rightarrow \mathcal{V}$  is said to be uniformly nonsingular if there exists  $\alpha > 0$  such that for any  $d_1, d_2, \dots, d_r \geq 0$  and  $d_{ij} \geq 0$ , any Jordan frame  $\{c_1, c_2, \dots, c_r\}$  and any  $x, y \in \mathcal{V}$ ,

$$\left\| f(x) - f(y) + \sum_{i=1}^r d_i(x_i - y_i)c_i + \sum_{i < j} d_{ij}(x_{ij} - y_{ij}) \right\| \geq \alpha \|x - y\|.$$

Note that the uniform nonsingularity is an extension of **P**-properties, i.e., if  $\mathcal{V} = \mathfrak{R}^n$  the uniform nonsingularity is equivalent to **P**-function property (see Proposition 4.1 of [21]). In [21], a Chen and Mangasarian smoothing method has been proposed for solving NLCP( $\psi, q$ ) where  $\psi$  is uniformly nonsingular. In [22], the authors showed several implications among the **Cartesian P** properties and uniform nonsingularity and discussed the existence of Newton directions and the boundedness of iterates of some merit function methods. The authors also showed that LCP( $L, q$ ) is globally uniquely solvable under the assumption of uniform nonsingularity.

Some geometrical properties on the solution set of the SCCP have been explored in [77, 78]. Let us denote by  $SOL(L, q)$  the solution set of LCP( $L, q$ ). A nonempty subset  $\mathcal{F}$  is said to be a *face* of a closed convex cone, denoted by  $\mathcal{K} \leq \mathcal{K}$ , if  $\mathcal{F}$  is a convex cone and

$$x \in \mathcal{K}, y - x \in \mathcal{K} \text{ and } y \in \mathcal{K} \implies x \in \mathcal{F}.$$

The *complementary face* of  $\mathcal{F}$  is defined as

$$\mathcal{F}^\Delta := \{y \in \mathcal{K}^* \mid \langle x, y \rangle = 0 \text{ for all } x \in \mathcal{F}\}.$$

Malik and Mohan [77] introduced the notion of the complementary cone which was originally introduced in [84] for  $\mathcal{K} = \mathfrak{R}_+^n$ .

**Definition 12.10 (Complementary cone of  $L$ ).** For a given linear operator  $L : \mathcal{V} \rightarrow \mathcal{V}$ , a complementary cone of  $L$  corresponding to the face  $\mathcal{F}$  of  $\mathcal{K}$  is defined as

$$C_{\mathcal{F}} := \{y - Lx \mid x \in \mathcal{F}, y \in \mathcal{F}^\Delta\}.$$

The linear complementarity problem LCP( $L, q$ ) has a solution if and only if there exists a face  $\mathcal{F}$  of  $\mathcal{K}$  such that  $q \in C_{\mathcal{F}}$  (see Observation 1 of [77]). Thus the union of all complementary cones is the set of all vectors  $q$  for which LCP( $L, q$ ) has a solution. Using these observations, the authors characterized the finiteness of the solution set of LCP( $L, q$ ).

## 12.6 Concluding Remarks

In this chapter, we provide a brief survey on the recent developments related to the symmetric cone complementarity problems. By viewing this, we strongly recognize the outstanding contribution of Paul Tseng to the area. We hope that this manuscript will help to leave his achievement for posterity.

**Acknowledgments** The author would like to thank an anonymous referee for valuable comments and suggestions which have greatly increased the clarity of this article.

## References

1. Alizadeh, A., Schmieta, S.H.: Symmetric cones, potential reduction methods and word-by-word extensions. In: Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.) *Handbook of semidefinite programming. Theory, algorithms, and applications*. Kluwer Academic Publishers, Boston, Dordrecht, London (2000)
2. Andersen, E., Ye, Y.: On a homogeneous algorithm for the monotone complementarity problems. *Math. Program.* **84**, 375–400 (1999)
3. Berman, A., Plemmons, R.J.: *Nonnegative Matrices in the Mathematical Sciences*. SIAM, Philadelphia, PA (1994)
4. Chang, Y.L., Pan, S., Chen, J.-S.: Strong semismoothness of Fischer-Burmeister complementarity function associated with symmetric cone. Preprint (2009)
5. Chi, X., Liu, S.: A one-step smoothing Newton method for second-order cone programming. *J. Comput. Appl. Math.* **223**, 114–123 (2009)
6. Chan, Z.X., Sun, D.: Constraint nondegeneracy, strong regularity, and nonsingularity in semidefinite programming. *SIAM J. Optim.* **19**, 370–396 (2008)
7. Chen, B., Harker, P.T.: A non-interior-point continuation method for linear complementarity problems. *SIAM J. Matrix Anal. Appl.* **14**, 1168–1190 (1993)
8. Che, B., Harker, P.T.: A continuation method for monotone variational inequalities. *Math. Program.* **69**, 237–253 (1995)
9. Chen, C., Mangasarian, O.L.: Smoothing methods for convex inequalities and linear complementarity problems. *Math. Program.* **71**, 51–69 (1995)
10. Chen, C., Mangasarian, O.L.: A class of smoothing functions for nonlinear and mixed complementarity problems. *Comput. Optim. Appl.* **5**, 97–138 (1996)
11. Chen, J.-S., Chen, X., Tseng, P.: Analysis of nonsmooth vector-valued functions associated with second-order cones. *Math. Program. Ser. B* **101**, 95–117 (2004)
12. Chen, J.-S., Tseng, P.: An unconstrained smooth minimization reformulation of the second-order cone complementarity problem. *Math. Program. Ser. B* **104**, 293–327 (2005)
13. Chen, J.-S.: Two classes of merit functions for the second-order cone complementarity problem. *Math. Methods Oper. Res.* **64**, 495–519 (2006)
14. Chen, J.-S.: A new merit function and its related properties for the second-order cone complementarity problem. *Pac. J. Optim.* **2**, 167–179 (2006)
15. Chen, J.-S.: Conditions for error bounds and bounded level sets of some merit functions for the second-order cone complementarity problem. *J. Optim. Theory Appl.* **135**, 459–473 (2007)
16. Chen, J.-S., Pan, S.: A descent method for a reformulation of the second-order cone complementarity problem. *J. Comput. Appl. Math.* **213**, 547–558 (2008)
17. Chen, X., Tseng, P.: Non-interior continuation methods for solving semidefinite complementarity problems. *Math. Program.* **95**, 431–474 (2003)
18. Chen, X., Qi, H.-D., Tseng, P.: Analysis of nonsmooth symmetric matrix functions with applications to semidefinite complementarity problems. *SIAM J. Optim.* **13**, 960–985 (2003)
19. Chen, X.D., Sun, D., Sun, J.: Complementarity functions and numerical experiments on some smoothing newton methods for second-order-cone complementarity problems. *Comput. Optim. Appl.* **25**, 39–56 (2003)
20. Chen, X., Qi, H.-D.: Cartesian  $\mathbf{P}$ -property and its applications to the semidefinite linear complementarity problem. *Math. Program.* **106**, 177–201 (2006)
21. Chua, C.B., Yi, P.: A continuation method for nonlinear complementarity problems over symmetric cone. *SIAM J. Optim.* **20**, 2560–2583 (2010)

22. Chua, C.B., Lin, H., Yi, P.: Uniform nonsingularity and complementarity problems over symmetric cones. Research report, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore (2009)
23. Clarke, F.H.: Optimization and Nonsmooth Analysis. John Wiley & Sons, New York (1983)
24. Cottle, R.W., Pang, J.-S., Stone, R.E.: The Linear Complementarity Problem. Academic Press, New York (1992)
25. Facchinei, F., Pang, J.-S.: Finite-Dimensional Variational Inequalities and Complementarity Problems Volume I, II. Springer-Verlag, New York, Berlin, Heidelberg (2003)
26. Fang, L., He, G., Huc, Y.: A new smoothing Newton-type method for second-order cone programming problems *Appl. Math. Comput.* **215**, 1020–1029 (2009)
27. Faraut, J., Korányi, A.: Analysis on Symmetric Cones. Oxford Science Publishers, New York (1994)
28. Faybusovich, L.: Euclidean Jordan algebras and interior-point algorithms. *Positivity* **1**, 331–357 (1997)
29. Fischer, A.: A special Newton-type optimization method. *Optim.* **24**, 269–284 (1992)
30. Fischer, A.: A Newton-type method for positive semidefinite linear complementarity problems. *J. Optim. Theory Appl.* **86**, 585–608 (1995)
31. Fischer, A.: On the local superlinear convergence of a Newton-type method for LCP under weak conditions. *Optim. Methods Software* **6**, 83–107 (1995)
32. Fukushima, N., Luo, Z.-Q., Tseng, P.: Smoothing functions for second-order-cone complementarity problems. *SIAM J. Optim.* **12**, 436–460 (2002)
33. Gowda, M.S., Song, Y.: On semidefinite linear complementarity problems. *Math. Program. Ser. A* **88**, 575–587 (2000)
34. Gowda, M.S., Song, Y.: Errata: On semidefinite linear complementarity problems. *Math. Program. Ser. A* **91**, 199–200 (2001)
35. Gowda, M.S., Sznajder, R., Tao, J.: Some **P**-properties for linear transformations on Euclidean Jordan algebras. Special issue on Positivity, *Linear Algebra Appl.* **393**, 203–232 (2004)
36. Gowda, M.S., Tao, J.: Some **P**-properties for nonlinear transformations on Euclidean Jordan algebras. *Math. Oper. Res.* **30**, 985–1004 (2005)
37. Gowda, M.S., Sznajder, R.: Automorphism invariance of **P** and **GUS** properties of linear transformations on Euclidean Jordan algebras. *Math. Oper. Res.* **31**, 109–123 (2006)
38. Gowda, M.S., Sznajder, R.: Some global uniqueness and solvability results for linear complementarity problems over symmetric cones. *SIAM J. Optim.* **18**, 461–481 (2007)
39. Gu, J., Zhang, L., Xiao, X.: Log-sigmoid nonlinear Lagrange method for nonlinear optimization problems over second-order cones. *J. Comput. Appl. Math.* **229**, 129–144 (2009)
40. Gurtuna, F., Petra, C., Potra, F.A., Shevchenko, O., Vancea, A.: Corrector-predictor methods for sufficient linear complementarity problems. *Comput Optim Appl.* (2009). doi:10.1007/s10589-009-9263-4
41. Han, D.: On the coerciveness of some merit functions for complementarity problems over symmetric cones. *J. Math. Anal. Appl.* **336**, 727–737 (2007)
42. Houser, R.A., Güler, O.: Self-scaled barrier functions on symmetric cones and their classification. *Found. Comput. Math.* **2**, 121–143 (2002)
43. Hayashi, S., Yamashita, N., Fukushima, M.: Robust Nash equilibria and second-order cone complementarity problems. *J. Nonlinear Convex Anal.* **6**, 283–296 (2005)
44. Hayashi, S., Yamashita, N., Fukushima, M.: A combined smoothing and regularization method for monotone second-order cone complementarity problems. *SIAM J. Optim.* **15**, 593–615 (2005)
45. Huang, Z.-H., Han, J.: Non-interior continuation method for solving the monotone semidefinite complementarity problem. *Appl. Math. Optim.* **47**, 195–211 (2003)
46. Huang, Z.-H.: Global Lipschitzian error bounds for semidefinite complementarity problems with emphasis on NCPs. *Appl. Math. Comput.* **162**, 1237–1258 (2005)
47. Huang, Z.-H., Ni, T.: Smoothing algorithms for complementarity problems over symmetric cones. *Comput. Optim. Appl.* **45**, 557–579 (2010)

48. Kanno, Y., Martins, J.A.C., Pinto da Costa, A.: Three-dimensional quasi-static frictional contact by using second-order cone linear complementarity problem. *Int. J. Numer. Methods Eng.* **65**, 62–83 (2006)
49. Kanzow, C.: Some noninterior continuation methods for linear complementarity problems. *SIAM J. Matrix Anal. Appl.* **17**, 851–868 (1996)
50. Kanzow, C.: A new approach to continuation methods for complementarity problems with uniform P-functions. *Oper. Res. Lett.* **20**, 85–92 (1997)
51. Kanzow, C., Nagel, C.: Quadratic convergence of a nonsmooth Newton-type method for semidefinite programs without strict complementarity. *SIAM J. Optim.* **15**, 654–672 (2005)
52. Kojima, M., Shindoh, S., Hara, S.: Interior-point methods for the monotone linear complementarity problem in symmetric matrices. *SIAM J. Optim.* **7**, 86–125 (1997)
53. Kojima, M., Shida, M., Shindoh, S.: Reduction of monotone linear complementarity problems over cones to linear programs over cones. *Acta Math. Vietnamica* **22**, 147–157 (1997)
54. Kojima, M., Shida, M., Shindoh, S.: Local convergence of predictor-corrector infeasible-interior-point method for SDPs and SDLCPs. *Math. Program.* **80**, 129–160 (1998)
55. Kojima, M., Shida, M., Shindoh, S.: A predictor-corrector interior-point algorithm for the semidefinite linear complementarity problem using the Alizadeh-Haeberly-Overton search direction. *SIAM J. Optim.* **9**, 444–465 (1999)
56. Kong, L., Xiu, N.: New smooth C-functions for symmetric cone complementarity problems. *Optim. Lett.* **1**, 391–400 (2007)
57. Kong, L., Tunçel, L., Xiu, N.: Monotonicity of Löwner operators and its applications to symmetric cone complementarity problems. Research Report CORR 2007-07, Department of Combinatorics and Optimization, University of Waterloo, Waterloo (2007)
58. Kong, L., Tunçel, L., Xiu, N.: The Fischer-Burmeister complementarity function on Euclidean Jordan algebras. *Pac. J. Optim.* **6**, 423–440 (2010)
59. Kong, L., Xiu, N., Han, J.: The solution set structure of monotone linear complementarity problems over second-order cone. *Oper. Res. Lett.* **36**, 71–76 (2008)
60. Kong, L., Sun, J., Xiu, N.: A regularized smoothing Newton method for symmetric cone complementarity problems. *SIAM J. Optim.* **19**, 1028–1047 (2008)
61. Kong, L., Tunçel, L., Xiu, N.: Vector-valued implicit Lagrangian for symmetric cone complementarity problems. *Asia-Pac. J. Oper. Res.* **26**, 199–233 (2009)
62. Kong, L., Tunçel, L., Xiu, N.: Clarke generalized Jacobian of the projection onto symmetric cones. *Set-Valued Variational Anal.* **17**, 135–151 (2009)
63. Kong, L., Tunçel, L., Xiu, N.: Homogeneous cone complementarity problems and *P* properties. Preprint (2009)
64. Korányi, A.: Monotone functions on formally real Jordan algebras. *Math. Ann.* **269**, 73–76 (1984)
65. Kum, S.H., Lim, Y.D.: Coercivity and strong semismoothness of the penalized Fischer-Burmeister function for the symmetric cone complementarity problem. *J. Optim. Theory Appl.* **142**, 377–383 (2009)
66. Kum, S.H., Lim, Y.D.: Penalized complementarity functions on symmetric cones. *J. Global Optim.* **46**, 475–485 (2010)
67. Li, Y.M., Wang, X.T., Wei, D.Y.: A new class of smoothing complementarity functions over symmetric cones. *Commun. Nonlinear Sci. Numer. Simul.* (2009). doi:10.1016/j.cnsns.2009.12.006
68. Liu, X.-H., Huang, Z.-H.: A smoothing Newton algorithm based on a one-parametric class of smoothing functions for linear programming over symmetric cones. *Math. Methods Oper. Res.* **70**, 385–404 (2009)
69. Liu, Y., Zhang, L., Liu, M.: Extension of smoothing functions to symmetric cone complementarity problems. *Appl. Math. J. Chin. Univ.* **22**, 245–252 (2007)
70. Löwner, K.: Über monotone matrixfunctionen. *Math. Z.* **38**, 177–216 (1934)
71. Lu, N., Huang, Z.-H., Han, J.: Properties of a class of nonlinear transformations over Euclidean Jordan algebras with applications to complementarity problems. *Numer. Funct. Anal. Optim.* **30**, 799–821 (2009)

72. Luo, X.-D., Tseng, P.: On a global projection-type error bound for the linear complementarity problem. *Linear Algebra Appl.* **253**, 251–278 (1997)
73. Lu, Z., Monteiro, R.D.C.: A note on the local convergence of a predictor-corrector interior-point algorithm for the semidefinite linear complementarity problem based on the Alizadeh-Haeberly-Overton search direction. *SIAM J. Optim.* **15**, 1147–1154 (2005)
74. Luo, Z., Xiu, N.: An  $O(rL)$  infeasible interior-point algorithm for symmetric cone LCP via CHKS function. *Acta Math. Appl. Sin. (English Series)* **25**, 593–606 (2009)
75. Luo, Z., Xiu, N.: Path-following interior point algorithms for the Cartesian  $\mathbf{P}_*(\kappa)$  LCP over symmetric cones. *Sci. China Ser. A: Math.* **52**, 1769–1784 (2009)
76. Luo, Z.-Q., Pang, J.-S.: Error bounds for analytic systems and their applications. *Math. Program.* **67**, 1–28 (1995)
77. Malik, M., Mohan, S.R.: Some geometrical aspects of semidefinite linear complementarity problems. *Linear and Multilinear Algebra* **54**, 55–70 (2006)
78. Malik, M., Mohan, S.R.: Cone complementarity problems with finite solution sets. *Oper. Res. Lett.* **34**, 121–126 (2006)
79. Mifflin, R.: Semismooth and semiconvex functions in constrained optimization. *SIAM J. Control Optim.* **15**, 957–972 (1977)
80. Mifflin, R.: A modification and an extension of Lemarechal's algorithm for nonsmooth minimization. *Math. Program. Stud.* **17**, 77–90 (1982)
81. Monteiro, R.D.C., Pang, J.-S.: On two interior-point mappings for nonlinear semidefinite complementarity problems. *Math. Oper. Res.* **23**, 39–60 (1998)
82. Monteiro, R.D.C., Zhang, Y.: A unified analysis for a class of path-following primal-dual interior-point algorithms for semidefinite programming. *Math. Program.* **81**, 281–299 (1998)
83. Muramatsu, M.: On commutative class of search directions for linear programming over symmetric cones. *J. Optim. Theory Appl.* **112**, 595–625 (2002)
84. Murty, K.G.: On the number of solutions to the complementarity problem and spanning properties of complementary cones. *Linear Algebra Appl.* **5**, 65–108 (1972)
85. Nesterov, Yu., Nemirovski, A.: *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM Studies in Applied Mathematics, SIAM, Philadelphia (1994)
86. Ni, T., Gu, W.-G.: Smoothing Newton algorithm for symmetric cone complementarity problems based on a one-parametric class of smoothing functions. *J. Appl. Math. Comput.* (2009). doi:10.1007/s12190-009-0341-7
87. Nishimura, R., Hayashi, S., Fukushima, M.: Robust Nash equilibria in N-person non-cooperative games: Uniqueness and reformulation. *Pac. J. Optim.* **5**, 237–259 (2009)
88. Outrata, J.V., Sun, D.: On the coderivative of the projection operator onto the second-order cone. *Set-Valued Anal.* **16**, 999–1014 (2008)
89. Pan, S., Chen, J.-S.: A one-parametric class of merit functions for the symmetric cone complementarity problem. *J. Math. Anal. Appl.* **355**, 195–215 (2009)
90. Pan, S., Chen, J.-S.: A regularization method for the second-order cone complementarity problem with the Cartesian  $P_0$ -property. *Nonlinear Anal.: Theory, Methods Appl.* **70**, 1475–1491 (2009)
91. Pan, S., Chen, J.-S.: A one-parametric class of merit functions for the second-order cone complementarity problem. *Comput. Optim. Appl.* **45**, 581–606 (2010)
92. Pan, S., Chen, J.-S.: Growth behavior of two classes of merit functions for symmetric cone complementarity problems. *J. Optim. Theory Appl.* **141**, 167–191 (2009)
93. Pan, S., Chen, J.-S.: A damped Gauss-Newton method for the second-order cone complementarity problem. *Appl. Math. Optim.* **59**, 293–318 (2009)
94. Pang, J.-S., Gabriel, S.A.: NE/SQP: A robust algorithm for the nonlinear complementarity problem. *Math. Program.* **60**, 295–337 (1993)
95. Potra, F.A., Sheng, R.: A superlinearly convergent primal-dual infeasible-interior-Point algorithm for semidefinite programming. *SIAM J. Optim.* **8**, 1007–1028 (1998)
96. Pan, S., Chen, J.-S.: Approximal gradient descent method for the extended second-order cone linear complementarity problem. *J. Math. Anal. Appl.* **366**, 164–180 (2010)

97. Potra, F.A.: Q-superlinear convergence of the iterates in primal-dual interior-point methods. *Math. Program.* **91**, 99–115 (2001)
98. Potra, F.A.: An infeasible interior point method for linear complementarity problems over symmetric cones. *AIP Conf. Proc.* **1168**, 1403–1406 (2009)
99. Preiß, M., Stoer, J.: Analysis of infeasible-interior-point paths arising with semidefinite linear complementarity problems. *Math. Program. Ser. A* **99**, 499–520 (2004)
100. Qin, L., Kong, L., Han, J.: Sufficiency of linear transformations on Euclidean Jordan algebras. *Optim. Lett.* **3**, 265–276 (2009)
101. Rangarajan, B.K.: Polynomial convergence of infeasible-interior-point methods over symmetric cones. *SIAM J. Optim.* **16**, 1211–1229 (2006)
102. Renegar, J.: *A Mathematical View of Interior Point Methods for Convex Optimization*. MPS-SIAM Series on Optimization, SIAM & MPS, Philadelphia (2001)
103. Rui, S.-P., Xu, C.-X.: Inexact non-interior continuation method for solving large-scale monotone SDCP. *Appl. Math. Comput.* **215**, 2521–2527 (2009)
104. Schmieta, S.H., Alizadeh, F.: Extension of primal-dual interior-point algorithm to symmetric cones. *Math. Program.* **96**, 409–438 (2003)
105. Shida, M., Shindoh, S., Kojima, M.: Centers of monotone generalized complementarity problems. *Math. Oper. Res.* **22**, 969–976 (1997)
106. Sim, C.-K., Zhao, G.: Underlying paths in interior point method for monotone semidefinite linear complementarity problem. *Math. Program.* **110**, 475–499 (2007)
107. Sim, C.-K., Zhao, G.: Asymptotic Behavior of HKM paths in interior point method for monotone semidefinite linear complementarity problem: General theory. *J. Optim. Theory Appl.* **137**, 11–25 (2008)
108. Sim, C.-K.: On the analyticity of underlying HKM paths for monotone semidefinite linear complementarity problem. *J. Optim. Theory Appl.* **141**, 193–215 (2009)
109. Sim, C.-K.: Superlinear convergence of an infeasible predictor-corrector path-following interior point algorithm for a semidefinite linear complementarity problem using the Helmberg-Kojima-Monteiro direction. *SIAM J. Optim.* **21**, 102–126 (2011)
110. Smale, S.: Algorithms for solving equations. In: Gleason, A.M. (ed.) *Proceedings of the International Congress of Mathematicians*, pp. 172–195. American Mathematical Society, Providence, Rhode Island (1987)
111. Sun, D., Sun, J.: Semismooth matrix valued functions. *Math. Oper. Res.* **27**, 150–169 (2002)
112. Sun, J., Sun, D., Qi, L.: A squared smoothing Newton method for nonsmooth matrix equations and its applications in semidefinite optimization problems. *SIAM J. Optim.* **14**, 783–806 (2003)
113. Sun, D., Sun, J.: Strong semismoothness of the Fischer-Burmeister SDC and SOC complementarity functions. *Math. Program.* **103**, 575–581 (2005)
114. Sun, D., Sun, J.: Löwner’s operator and spectral functions in Euclidean Jordan algebras. *Math. Oper. Res.* **33**, 421–445 (2008)
115. Sun, J., Zhan, L.: A globally convergent method based on Fischer-Burmeister operators for solving second-order cone constrained variational inequality problems. *Comput. Math. Appl.* **58**, 1936–1946 (2009)
116. Tao, J., Gowda, M.S.: Some **P**-properties for nonlinear transformations on Euclidean Jordan algebras. *Math. Oper. Res.* **30**, 985–1004 (2005)
117. Tao, J.: Positive principal minor property of linear transformations on Euclidean Jordan algebras. *J. Optim. Theory Appl.* **140**, 131–152 (2009)
118. Tao, J.: Strict semimonotonicity property of linear transformations on Euclidean Jordan algebras. *J. Optim. Theory Appl.* **144**, 575–596 (2010)
119. Tseng, P.: Merit functions for semi-definite complementarity problems. *Math. Program.* **83**, 159–185 (1998)
120. Tseng, P.: Analysis of a non-interior continuation method based on Chen-Mangasarian smoothing functions for complementarity problems. In: Fukushima, M., Qi, L. (eds.) *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods*, pp. 381–404. Kluwer Academic Publishers, Boston (1998)

121. Xu, S., Burke, J.V.: A polynomial time interior-point path-following algorithm for LCP based on Chen-Harker-Kanzow smoothing techniques. *Math. Program.* **86**, 91–103 (1999)
122. Yamashita, N., Fukushima, M.: A new merit function and a descent method for semidefinite complementarity problems. In: Fukushima, M., Qi, Y. (eds.) *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods*, pp. 405–420. Kluwer Academic Publishers, Boston (1998)
123. Yan, T., Fukushima, M.: Smoothing method for mathematical programs with symmetric cone complementarity constraints. Technical Report 2009-017, Department of Applied Mathematics and Physics, Kyoto University (2009)
124. Yoshise, A.: Interior point trajectories and a homogeneous model for nonlinear complementarity problems over symmetric cones. *SIAM J. Optim.* **17**, 1129–1153 (2006)
125. Yoshise, A.: Homogeneous algorithms for monotone complementarity problems over symmetric cones. *Pac. J. Optim.* **5**, 313–337 (2009)
126. Zhang, X., Liu, S., Liu, Z.: A smoothing method for second order cone complementarity problem. *J. Comput. Appl. Math.* **22**, 83–91 (2009)
127. Yu, Z.: The bounded smooth reformulation and a trust region algorithm for semidefinite complementarity problems. *Appl. Math. Comput.* **159**, 157–170 (2004)

# Chapter 13

## Convexity and Semidefinite Programming in Dimension-Free Matrix Unknowns

J. William Helton, Igor Klep, and Scott McCullough

### 13.1 Introduction

Given symmetric  $\ell \times \ell$  symmetric matrices  $A_j$  with real entries, the expression

$$L(x) = I_\ell + \sum_{j=1}^g A_j x_j > 0 \quad (13.1)$$

is a **linear matrix inequality** (LMI). Here  $> 0$  means positive definite,  $x = (x_1, \dots, x_g) \in \mathbb{R}^g$  and of interest is the set of solutions  $x$ . Taking advantage of the Kronecker (tensor) product  $A \otimes B$  of matrices, it is natural to consider, for tuples of symmetric  $n \times n$  matrices  $X = (X_1, \dots, X_g) \in (\mathbb{SR}^{n \times n})^g$ , the inequality

$$L(X) = I_\ell \otimes I_n + \sum_{j=1}^g A_j \otimes X_j > 0. \quad (13.2)$$

---

J.W. Helton (✉)

Department of Mathematics, University of California at San Diego, 9500 Gilman Drive,  
La Jolla, CA 92093-0112, USA  
e-mail: [helton@math.ucsd.edu](mailto:helton@math.ucsd.edu)

I. Klep

Univerza v Ljubljani, Fakulteta za matematiko in fiziko, Jadranska 21,  
1111 Ljubljana, Slovenija

Univerza v Mariboru, Fakulteta za naravoslovje in matematiko, Koroška 160,  
2000 Maribor, Slovenija  
e-mail: [igor.klep@fmf.uni-lj.si](mailto:igor.klep@fmf.uni-lj.si)

S. McCullough

Department of Mathematics, University of Florida, 490 Little Hall, Gainesville,  
FL 32611-8105, USA  
e-mail: [sam@math.ufl.edu](mailto:sam@math.ufl.edu)

For reasons which will become apparent soon, we call expression (13.2) a **non-commutative LMI** (nc LMI). Letting  $\mathcal{D}_L(n)$  denote the solutions  $X$  of size  $n \times n$ , note that  $\mathcal{D}_L(1)$  is the solution set of (13.1). In many areas of mathematics and its applications, the inequality (13.2) is called the **quantized** version of inequality (13.1).

Quantizing a polynomial inequality requires the notion of a non-commutative (free) polynomial which can loosely be thought of as a polynomial in matrix unknowns. Section 13.1.2 below gives the details on these polynomials. For now we limit the discussion to the example,

$$p(x, y) = 4 - x - y - (2x^2 + xy + yx + 2y^2). \quad (13.3)$$

Of course, for symmetric  $n \times n$  matrices  $X, Y$ ,

$$p(X, Y) = 4I_n - X - Y - (2X^2 + XY + YX + 2Y^2). \quad (13.4)$$

The set  $\{(x, y) \in \mathbb{R}^2 : p(x, y) > 0\}$  is a semi-algebraic set. By analogy, the set  $\{(X, Y) : p(X, Y) > 0\}$  is a **non-commutative semi-algebraic set**.

nc LMIs, and more generally non-commutative semi-algebraic sets, arise naturally in semidefinite programming (SDP) and in linear systems theory problems determined by a signal-flow diagram. They are of course basic objects in the study of operator spaces and thus are related to problems like Connes' embedding conjecture [12, 31] and the Bessis–Moussa–Villani (BMV) conjecture [8] from quantum statistical mechanics [32]. As is seen in Theorem 13.4 below, they even have something to say about their scalar (commutative) counterparts. For some these non-commutative considerations have their own intrinsic interest as a free analog to classical semi-algebraic geometry.

Non-commutative will often be shortened to nc.

### 13.1.1 The Roadmap

The chapter treats four areas of research concerning nc LMIs and nc polynomials.

In the remainder of the introduction we first, in Sect. 13.1.2, give additional background on a core object of our study, polynomials in non-commuting variables. The initiated reader may wish to skip this subsection. Sections 13.1.3–13.1.6 give overviews of the four main topics of the survey.

The body of the chapter consists of six sections. The first four give further detail on the main topics. Except for Sect. 13.3 which has its own motivation subsection, motivation for our investigations is weaved into the discussion. Convexity is a recurring theme. Section 13.6 offers a list of computer algebra packages for work in a free  $*$ -algebra revolving around convexity and positivity.

### 13.1.2 Non-commutative Polynomials

Let  $\mathbb{R}\langle x \rangle$  denote the real algebra of polynomials in the non-commuting indeterminates  $x = (x_1, \dots, x_g)$ . Elements of  $\mathbb{R}\langle x \rangle$  are **non-commutative polynomials**, abbreviated to **nc polynomials** or often just **polynomials**. Thus, a non-commutative polynomial  $p$  is a finite sum,

$$p = \sum p_w w, \quad (13.5)$$

where each  $w$  is a word in  $(x_1, \dots, x_g)$  and the coefficients  $p_w \in \mathbb{R}$ . The polynomial  $p$  of (13.3) is a non-commutative polynomial of degree two in two variables. The polynomial

$$q = x_1 x_2^3 + x_2^3 x_1 + x_3 x_1 x_2 - x_2 x_1 x_3 \quad (13.6)$$

is a non-commutative polynomial of degree four in three variables.

#### 13.1.2.1 Involution

There is a natural **involution** \* on  $\mathbb{R}\langle x \rangle$  given by

$$p^* = \sum p_w w^*, \quad (13.7)$$

where, for a word  $w$ ,

$$w = x_{j_1} x_{j_2} \cdots x_{j_n} \mapsto w^* = x_{j_n} \cdots x_{j_2} x_{j_1}. \quad (13.8)$$

A polynomial  $p$  is **symmetric** if  $p^* = p$ . For example, the polynomial  $p$  of (13.3) is symmetric, whereas the  $q$  of (13.6) is not. In particular,  $x_j^* = x_j$  and for this reason the variables are sometimes referred to as symmetric non-commuting variables.

Denote, by  $\mathbb{R}\langle x \rangle_d$ , the polynomials in  $\mathbb{R}\langle x \rangle$  of (total) degree  $d$  or less.

#### 13.1.2.2 Substituting Matrices for Indeterminates

Let  $(\mathbb{SR}^{n \times n})^g$  denote the set of  $g$ -tuples  $X = (X_1, \dots, X_g)$  of real symmetric  $n \times n$  matrices. A polynomial  $p(x) = p(x_1, \dots, x_g) \in \mathbb{R}\langle x \rangle$  can naturally be evaluated at a tuple  $X \in (\mathbb{SR}^{n \times n})^g$  resulting in an  $n \times n$  matrix. Equations (13.3) and (13.4) are illustrative. In particular, the constant term  $p_\emptyset$  of  $p(x)$  becomes  $p_\emptyset I_n$ ; i.e., the empty word evaluates to  $I_n$ . Often we write  $p(0)$  for  $p_\emptyset$  interpreting the 0 as  $0 \in \mathbb{R}^g$ . As a further example, for the polynomial  $q$  from (13.6),

$$q(X) = X_1 X_2^3 - X_2^3 X_1 + X_3 X_1 X_2 + X_2 X_1 X_3.$$

The involution on  $\mathbb{R}\langle x \rangle$  that was introduced earlier is compatible with evaluation at  $X$  and matrix transposition, i.e.,

$$p^*(X) = p(X)^*,$$

where  $p(X)^*$  denotes the transpose of the matrix  $p(X)$ . Note, if  $p$  is symmetric, then so is  $p(X)$ .

### 13.1.2.3 Matrix-Valued Polynomials

Let  $\mathbb{R}\langle x \rangle^{\delta \times \delta'}$  denote the  $\delta \times \delta'$  matrices with entries from  $\mathbb{R}\langle x \rangle$ . In particular, if  $p \in \mathbb{R}\langle x \rangle^{\delta \times \delta'}$ , then

$$p = \sum p_w w, \quad (13.9)$$

where the sum is finite and each  $p_w$  is a real  $\delta \times \delta'$  matrix. Denote, by  $\mathbb{R}\langle x \rangle_d^{\delta \times \delta'}$ , the subset of  $\mathbb{R}\langle x \rangle^{\delta \times \delta'}$  whose polynomial entries have degree  $d$  or less.

Evaluation at  $X \in (\mathbb{SR}^{n \times n})^g$  naturally extends to  $p \in \mathbb{R}\langle x \rangle^{\delta \times \delta'}$  via the Kronecker tensor product, with the result,  $p(X)$ , a  $\delta \times \delta'$  block matrix with  $n \times n$  entries. The involution  $*$  naturally extends to  $\mathbb{R}\langle x \rangle^{\delta \times \delta}$  by

$$p = \sum p_w^* w^*, \quad (13.10)$$

for  $p$  given by (13.9). A polynomial  $p \in \mathbb{R}\langle x \rangle^{\delta \times \delta}$  is symmetric if  $p^* = p$  and in this case  $p(X) = p(X)^*$ .

A simple method of constructing new matrix valued polynomials from old ones is by direct sum. For instance, if  $p_j \in \mathbb{R}\langle x \rangle^{\delta_j \times \delta_j}$  for  $j = 1, 2$ , then

$$p_1 \oplus p_2 = \begin{bmatrix} p_1 & 0 \\ 0 & p_2 \end{bmatrix} \in \mathbb{R}\langle x \rangle^{(\delta_1 + \delta_2) \times (\delta_1 + \delta_2)}.$$

### 13.1.2.4 Linear Matrix Inequalities

Given symmetric  $\ell \times \ell$  matrices  $A_0, A_1, \dots, A_g$ , the expression

$$L(x) = A_0 + \sum_{j=1}^g A_j x_j \quad (13.11)$$

is an affine linear nc matrix polynomial, better known as a **linear** (or affine linear) **pencil**. In the case that  $A_0 = 0$ ,  $L$  is a **truly linear pencil**; and when  $A_0 = I$ , we say  $L$  is a **monic linear pencil**.

The inequality  $L(x) > 0$  for  $x \in \mathbb{R}^g$  is a **linear matrix inequality (LMI)**. LMIs are ubiquitous in science and engineering. Evaluation of  $L$  at  $X \in (\mathbb{SR}^{n \times n})^g$  is most easily described using tensor products as in (13.2) and the expression  $L(X) > 0$  is a **non-commutative LMI**, or nc LMI for short.

### 13.1.3 LMI Domination and Complete Positivity

This section discusses the nc LMI versions of two natural LMI domination questions. To fix notation, let

$$L(x) = A_0 + \sum_{j=1}^g A_j x_j,$$

be a given linear pencil (thus the  $A_j$  are symmetric  $\ell \times \ell$  matrices). For a fixed  $n$ , the solution set of all  $X \in (\mathbb{SR}^{n \times n})^g$  satisfying  $L(X) > 0$  is denoted  $\mathcal{D}_L(n)$  and the sequence (graded set)  $(\mathcal{D}_L(n))_{n \in \mathbb{N}}$  is written  $\mathcal{D}_L$ . Note that  $\mathcal{D}_L(1)$  is the solution set of the classical (commutative) LMI,  $L(x) > 0$ .

Given linear matrix inequalities (LMIs)  $L_1$  and  $L_2$  it is natural to ask:

- (Q<sub>1</sub>) when does one dominate the other, that is, when is  $\mathcal{D}_{L_1}(1) \subseteq \mathcal{D}_{L_2}(1)$ ?
- (Q<sub>2</sub>) when are they mutually dominant, that is,  $\mathcal{D}_{L_1}(1) = \mathcal{D}_{L_2}(1)$ ?

While such problems can be NP-hard, their nc relaxations have elegant answers. Indeed, they reduce to constructible semidefinite programs. We chose to begin with this topic because it offers the most gentle introduction to our matrix subject.

To describe a sample result, assume there is an  $x \in \mathbb{R}^g$  such that  $L_1(x)$  and  $L_2(x)$  are both positive definite, and suppose  $\mathcal{D}_{L_1}(1)$  is bounded. If  $\mathcal{D}_{L_1}(n) \subseteq \mathcal{D}_{L_2}(n)$  for every  $n$ , then there exist matrices  $V_j$  such that

$$L_2(x) = V_1^* L_1(x) V_1 + \cdots + V_\mu^* L_1(x) V_\mu. \quad (\text{A1})$$

The converse is of course immediate. As for (Q<sub>2</sub>) we show that  $L_1$  and  $L_2$  are mutually dominant ( $\mathcal{D}_{L_1}(n) = \mathcal{D}_{L_2}(n)$  for all  $n$ ) if and only if, up to certain redundancies described in detail in Sect. 13.2,  $L_1$  and  $L_2$  are unitarily equivalent.

It turns out that our matrix variable LMI domination problem is equivalent to the classical problem of determining if a linear map  $\tau$  from one subspace of matrices to another is “completely positive”. Complete positivity is one of the main techniques of modern operator theory and the theory of operator algebras. On one hand it provides tools for studying LMIs and on the other hand, since completely positive maps are not so far from representations and generally are more tractable than their merely positive counterparts, the theory of completely positive maps provides perspective on the difficulties in solving LMI domination problems. nc LMI domination is the topic of Sect. 13.2.

### 13.1.4 Non-commutative Convex Sets and LMI Representations

Section 13.1.3 dealt with the (matricial) solution set of a Linear Matrix Inequality

$$\mathcal{D}_L = \{X : L(X) > 0\}.$$

The set  $\mathcal{D}_L$  is convex in the sense that each  $\mathcal{D}_L(n)$  is convex. It is also a non-commutative basic open semi-algebraic set (in a sense we soon define). The main theorem of this section is the converse, a result which has implications for both semidefinite programming and systems engineering.

Let  $p \in \mathbb{R}\langle x \rangle^{\delta \times \delta}$  be a given symmetric non-commutative  $\delta \times \delta$ -valued matrix polynomial. Assuming that  $p(0) > 0$ , the positivity set  $\mathcal{D}_p(n)$  of a non-commutative symmetric polynomial  $p$  in dimension  $n$  is the component of 0 of the set

$$\{X \in (\mathbb{SR}^{n \times n})^g : p(X) > 0\}.$$

The **positivity set**,  $\mathcal{D}_p$ , is the sequence of sets  $(\mathcal{D}_p(n))$ , which is the type of set we call a **non-commutative basic open semi-algebraic set**. The non-commutative set  $\mathcal{D}_p$  is called **convex** if, for each  $n$ ,  $\mathcal{D}_p(n)$  is convex. A set is said to have a **Linear Matrix Inequality Representation** if it is the set of all solutions to some LMI, that is, it has the form  $\mathcal{D}_L$  for some  $L(x) = I + \sum_j A_j x_j$ .

The main theorem of Sect. 13.3 says: if  $p(0) > 0$  and  $\mathcal{D}_p$  is bounded, then  $\mathcal{D}_p$  has an LMI representation if and only if  $\mathcal{D}_p$  is convex.

### 13.1.5 Non-commutative Convex Polynomials Have Degree Two

We turn now from non-commutative convex sets to non-commutative convex polynomials. The previous section exposed the rigid the structure of sets which are both convex and the sublevel set of a non-commutative polynomial. Of course if  $p$  is concave ( $-p$  is convex), then its sublevel sets are convex. But more is true.

A symmetric polynomial  $p$  is **matrix convex**, if for each positive integer  $n$ , each pair of tuples of symmetric matrices  $X \in (\mathbb{SR}^{n \times n})^g$  and  $Y \in (\mathbb{SR}^{n \times n})^g$ , and each  $0 \leq t \leq 1$ ,

$$p(tX + (1-t)Y) \leq tp(X) + (1-t)p(Y).$$

The main result on convex polynomials, given in Sect. 13.4, is that every symmetric non-commutative polynomial which is matrix convex has degree two or less.

### 13.1.6 Algebraic Certificates of Non-commutative Positivity: Positivstellensätze

An algebraic certificate for positivity of a polynomial  $p$  on a semi-algebraic set  $S$  is a Positivstellensatz. The familiar fact that a polynomial  $p$  in one-variable which is positive on  $S = \mathbb{R}$  is a sum of squares is an example.

The theory of Positivstellensätze – a pillar of the field of semi-algebraic geometry – underlies the main approach currently used for global optimization of polynomials. See [34, 38] for a beautiful treatment of this, and other, applications of commutative semi-algebraic geometry. Further, because convexity of a polynomial  $p$  on a set  $S$  is equivalent to positivity of the Hessian of  $p$  on  $S$ , this theory also provides a link between convexity and semi-algebraic geometry. Indeed, this link in the non-commutative setting ultimately leads to the conclusion that a matrix convex non-commutative polynomial has degree at most two.

Polynomial optimization problems involving non-commuting variables also arise naturally in many areas of quantum physics, see [37, 42].

Positivstellensätze in various incarnations appear throughout this survey as they arise naturally in connection with the previous topics. Section 13.5 contains a brief list of algebraic certificates for positivity like conditions for non-commutative polynomials in both symmetric and non-symmetric nc variables. Thus, this section provides an overview of non-commutative semi-algebraic geometry with the theme being that nc Positivstellensätze are cleaner and more rigid than their commutative counterparts.

## 13.2 LMI Domination and Complete Positivity

In this section we expand upon the discussion of nc LMI domination of Sect. 13.1.3. Recall, a monic linear pencil is an expression of the form

$$L(x) = I + \sum_{j=1}^g A_j x_j,$$

where, for some  $\ell$ , the  $A_j$  are symmetric  $\ell \times \ell$  matrices with real entries and  $I$  is the  $\ell \times \ell$  identity. For a given positive integer  $n$ , let

$$\mathcal{D}_L(n) = \{X \in (\mathbb{SR}^{n \times n})^g : L(X) > 0\}$$

and let  $\mathcal{D}_L$  denote the sequence of sets  $(\mathcal{D}_L(n))_{n \in \mathbb{N}}$ . Thus  $\mathcal{D}_L$  is the solution set of the nc LMI  $L(X) > 0$  and  $\mathcal{D}_L(1)$  is the solution set of the traditional LMI  $L(x) > 0$  ( $x \in \mathbb{R}^g$ ). We call  $\mathcal{D}_L$  an **nc LMI**.

### 13.2.1 Certificates for LMI Domination

This subsection contains precise algebraic characterizations of nc LMI domination. Algorithms, the connection to complete positivity, examples, and the application to a new commutative Positivstellensatz follow in succeeding subsections.

**Theorem 13.1 (Linear Positivstellensatz [18]).** *Let  $L_j \in \mathbb{SR}^{d_j \times d_j}(x)$ ,  $j = 1, 2$ , be monic linear pencils and assume  $\mathcal{D}_{L_1}(1)$  is bounded. Then  $\mathcal{D}_{L_1} \subseteq \mathcal{D}_{L_2}$  if and only if there is a  $\mu \in \mathbb{N}$  and an isometry  $V \in \mathbb{R}^{\mu d_1 \times d_2}$  such that*

$$L_2(x) = V^*(I_\mu \otimes L_1(x))V = \sum_{j=1}^{\mu} V_j^* L_1(x) V_j. \quad (13.12)$$

Suppose  $L \in \mathbb{SR}^{d \times d}(x)$ ,

$$L = I + \sum_{j=1}^g A_j x_j$$

is a monic linear pencil. A subspace  $\mathcal{H} \subseteq \mathbb{R}^d$  is **reducing for  $L$**  if  $\mathcal{H}$  reduces each  $A_j$ ; i.e., if  $A_j \mathcal{H} \subseteq \mathcal{H}$ . Since each  $A_j$  is symmetric, it also follows that  $A_j \mathcal{H}^\perp \subseteq \mathcal{H}^\perp$ . Hence, with respect to the decomposition  $\mathbb{R}^d = \mathcal{H} \oplus \mathcal{H}^\perp$ ,  $L$  can be written as the direct sum,

$$L = \tilde{L} \oplus \tilde{L}^\perp = \begin{bmatrix} \tilde{L} & 0 \\ 0 & \tilde{L}^\perp \end{bmatrix} \quad \text{where} \quad \tilde{L} = I + \sum_{j=1}^g \tilde{A}_j x_j,$$

and  $\tilde{A}_j$  is the restriction of  $A_j$  to  $\mathcal{H}$ . (The pencil  $\tilde{L}^\perp$  is defined similarly.) If  $\mathcal{H}$  has dimension  $\ell$ , then by identifying  $\mathcal{H}$  with  $\mathbb{R}^\ell$ , the pencil  $\tilde{L}$  is a monic linear pencil of size  $\ell$ . We say that  $\tilde{L}$  is a **subpencil** of  $L$ . If moreover,  $\mathcal{D}_L = \mathcal{D}_{\tilde{L}}$ , then  $\tilde{L}$  is a **defining subpencil** and if no proper subpencil of  $\tilde{L}$  is a defining subpencil for  $\mathcal{D}_L$ , then  $\tilde{L}$  is a **minimal defining (sub)pencil**.

**Theorem 13.2 (Linear Gleichstellensatz [18]).** *Suppose  $L_1, L_2$  are monic linear pencils with  $\mathcal{D}_{L_1}(1)$  bounded. Then  $\mathcal{D}_{L_1} = \mathcal{D}_{L_2}$  if and only if minimal defining pencils  $\tilde{L}_1$  and  $\tilde{L}_2$  for  $\mathcal{D}_{L_1}$  and  $\mathcal{D}_{L_2}$  respectively, are unitarily equivalent. That is, there is a unitary matrix  $U$  such that*

$$\tilde{L}_2(x) = U^* \tilde{L}_1(x) U. \quad (13.13)$$

### 13.2.2 Algorithms for LMIs

Of widespread interest is determining if

$$\mathcal{D}_{L_1}(1) \subseteq \mathcal{D}_{L_2}(1), \quad (13.14)$$

or if  $\mathcal{D}_{L_1}(1) = \mathcal{D}_{L_2}(1)$ . For example, the paper of Ben-Tal and Nemirovski [9] exhibits simple cases where determining this is NP-hard. While we do not give details here we guide the reader to [18, Sect. 4] where we prove that  $\mathcal{D}_{L_1} \subseteq \mathcal{D}_{L_2}$  is equivalent to the feasibility of a certain semidefinite program which we construct explicitly in [18, Sect. 4.1]. Of course, if  $\mathcal{D}_{L_1} \subseteq \mathcal{D}_{L_2}$ , then  $\mathcal{D}_{L_1}(1) \subseteq \mathcal{D}_{L_2}(1)$ . Thus our algorithm is a type of relaxation of the problem (13.14).

Also in [18] is an algorithm (Sect. 4.2) easily adapted from the first to determine if  $\mathcal{D}_L$  is bounded, and what its “radius” is. By [18, Proposition 2.4],  $\mathcal{D}_L$  is bounded if and only if  $\mathcal{D}_L(1)$  is bounded. Our algorithm thus yields an upper bound of the radius of  $\mathcal{D}_L(1)$ . In [18, Sect. 4.3] we solve a matricial relaxation of the classical matrix cube problem, finding the biggest matrix cube contained in  $\mathcal{D}_L$ . Finally, given a matricial LMI set  $\mathcal{D}_L$ , [18, Sect. 4.4] gives an algorithm to compute the linear pencil  $\tilde{L} \in \mathbb{SR}^{d \times d}\langle x \rangle$  with smallest possible  $d$  satisfying  $\mathcal{D}_L = \mathcal{D}_{\tilde{L}}$ .

### 13.2.3 Complete Positivity and LMI Inclusion

To monic linear pencils  $L_1$  and  $L_2$ ,

$$L_j(x) = I + \sum_{\ell=1}^g A_{j,\ell} x_\ell \in \mathbb{SR}^{d_j \times d_j}\langle x \rangle, \quad j = 1, 2 \quad (13.15)$$

are the naturally associated subspaces of  $d_j \times d_j$  ( $j = 1, 2$ ) matrices

$$\mathcal{S}_j = \text{span}\{I, A_{j,\ell}: \ell = 1, \dots, g\} = \text{span}\{L_j(X): X \in \mathbb{R}^g\} \subseteq \mathbb{SR}^{d_j \times d_j}. \quad (13.16)$$

We shall soon see that the condition  $L_2$  dominates  $L_1$ , equivalently  $\mathcal{D}_{L_1} \subseteq \mathcal{D}_{L_2}$ , is equivalent to a property called complete positivity, defined below, of the unital linear mapping  $\tau: \mathcal{S}_1 \rightarrow \mathcal{S}_2$  determined by

$$\tau(A_{1,\ell}) = A_{2,\ell}. \quad (13.17)$$

A recurring theme in the non-commutative setting, such as that of a subspace of  $C^*$ -algebra [1–3] or in free probability [50, 51] to give two of many examples, is the need to consider the **complete matrix structure** afforded by tensoring with  $n \times n$  matrices (over positive integers  $n$ ). The resulting theory of operator algebras, systems, spaces and matrix convex sets has matured to the point that there are now several excellent books on the subject including [7, 39, 41].

Let  $\mathcal{T}_j \subseteq \mathbb{R}^{d_j \times d_j}$  be unital linear subspaces closed under the transpose, and  $\phi : \mathcal{T}_1 \rightarrow \mathcal{T}_2$  a unital linear  $*$ -map. For  $n \in \mathbb{N}$ ,  $\phi$  induces the map

$$\phi_n = I_n \otimes \phi : \mathbb{R}^{n \times n} \otimes \mathcal{T}_1 = \mathcal{T}_1^{n \times n} \rightarrow \mathcal{T}_2^{n \times n}, \quad M \otimes A \mapsto M \otimes \phi(A),$$

called an **ampliation** of  $\phi$ . Equivalently,

$$\phi_n \begin{pmatrix} T_{11} & \cdots & T_{1n} \\ \vdots & \ddots & \vdots \\ T_{n1} & \cdots & T_{nn} \end{pmatrix} = \begin{pmatrix} \phi(T_{11}) & \cdots & \phi(T_{1n}) \\ \vdots & \ddots & \vdots \\ \phi(T_{n1}) & \cdots & \phi(T_{nn}) \end{pmatrix}$$

for  $T_{ij} \in \mathcal{T}_1$ . We say that  $\phi$  is  **$k$ -positive** if  $\phi_k$  is a positive map. If  $\phi$  is  $k$ -positive for every  $k \in \mathbb{N}$ , then  $\phi$  is **completely positive**.

### 13.2.4 The Map $\tau$ Is Completely Positive

A basic observation is that  $n$ -positivity of  $\tau$  is equivalent to the inclusion  $\mathcal{D}_{L_1}(n) \subseteq \mathcal{D}_{L_2}(n)$ . Hence  $\mathcal{D}_{L_1} \subseteq \mathcal{D}_{L_2}$  is equivalent to complete positivity of  $\tau$ , an observation which ultimately leads to the algebraic characterization of Theorem 13.1.

**Theorem 13.3.** Consider the monic linear pencils of (13.15) and assume that  $\mathcal{D}_{L_1}(1)$  is bounded. Let  $\tau : \mathcal{S}_1 \rightarrow \mathcal{S}_2$  be the unital linear map of (13.17).

1.  $\tau$  is  $n$ -positive if and only if  $\mathcal{D}_{L_1}(n) \subseteq \mathcal{D}_{L_2}(n)$ ;
2.  $\tau$  is completely positive if and only if  $\mathcal{D}_{L_1} \subseteq \mathcal{D}_{L_2}$ .

Conversely, suppose  $\mathcal{D}$  is a unital  $*$ -subspace of  $\mathbb{SR}^{d \times d}$  and  $\tau : \mathcal{D} \rightarrow \mathbb{SR}^{d' \times d'}$  is completely positive. Given a basis  $\{I, A_1, \dots, A_g\}$  for  $\mathcal{D}$ , let  $B_j = \tau(A_j)$ . Let

$$L_1(x) = I + \sum A_j x_j, \quad L_2(x) = I + \sum B_j x_j.$$

The complete positivity of  $\tau$  implies, if  $L_1(X) > 0$ , then  $L_2(X) > 0$  and hence  $\mathcal{D}_{L_1} \subseteq \mathcal{D}_{L_2}$ . Hence the completely positive map  $\tau$  (together with a choice of basis) gives rise to an LMI domination.

### 13.2.5 An Example

The following example illustrates the constructs of the previous two subsections. Let

$$L_1(x_1, x_2) = I + \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} x_1 + \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} x_2 = \begin{bmatrix} 1 & x_1 & x_2 \\ x_1 & 1 & 0 \\ x_2 & 0 & 1 \end{bmatrix} \in \mathbb{SR}^{3 \times 3}\langle x \rangle$$

and

$$L_2(x_1, x_2) = I + \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} x_1 + \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} y_2 = \begin{bmatrix} 1+x_1 & x_2 \\ x_2 & 1-x_1 \end{bmatrix} \in \mathbb{SR}^{2 \times 2}\langle x \rangle.$$

Then

$$\begin{aligned} \mathcal{D}_{L_1} &= \{(X_1, X_2) : 1 - X_1^2 - X_2^2 > 0\}, \\ \mathcal{D}_{L_1}(1) &= \{(X_1, X_2) \in \mathbb{R}^2 : X_1^2 + X_2^2 < 1\}, \\ \mathcal{D}_{L_2}(1) &= \{(X_1, X_2) \in \mathbb{R}^2 : X_1^2 + X_2^2 < 1\}. \end{aligned}$$

Thus  $\mathcal{D}_{L_1}(1) = \mathcal{D}_{L_2}(1)$ . On one hand,

$$\left( \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & \frac{3}{4} \\ \frac{3}{4} & 0 \end{bmatrix} \right) \in \mathcal{D}_{L_1} \setminus \mathcal{D}_{L_2},$$

so  $L_1(X_1, X_2) > 0$  does not imply  $L_2(X_1, X_2) > 0$ .

On the other hand,  $L_2(X_1, X_2) > 0$  does imply  $L_1(X_1, X_2) > 0$ . The map  $\tau : \mathcal{S}_2 \rightarrow \mathcal{S}_1$  in our example is given by

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mapsto \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \mapsto \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \mapsto \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Consider the extension of  $\tau$  to a unital linear  $*$ -map  $\psi : \mathbb{R}^{2 \times 2} \rightarrow \mathbb{R}^{3 \times 3}$ , defined by

$$\begin{aligned} E_{11} &\mapsto \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad E_{12} \mapsto \frac{1}{2} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}, \\ E_{21} &\mapsto \frac{1}{2} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}, \quad E_{22} \mapsto \frac{1}{2} \begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned}$$

(Here  $E_{ij}$  are the  $2 \times 2$  matrix units.) To show that  $\psi$  is completely positive compute its Choi matrix defined as

$$C = \begin{bmatrix} \psi(E_{11}) & \psi(E_{12}) \\ \psi(E_{21}) & \psi(E_{22}) \end{bmatrix}. \quad (13.18)$$

[39, Theorem 3.14] says  $\psi$  is completely positive if and only if  $C \succeq 0$ . The Choi matrix is the key to computational algorithms in [18, Sect. 4]. In the present case, to see that  $C$  is positive semidefinite, note

$$C = \frac{1}{2} W^* W \quad \text{for} \quad W = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & -1 & 0 \end{bmatrix}.$$

Now  $\psi$  has a very nice representation:

$$\psi(S) = \frac{1}{2}V_1^*SV_1 + \frac{1}{2}V_2^*SV_2 = \frac{1}{2}\begin{bmatrix} V_1 \\ V_2 \end{bmatrix}^*\begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix}\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \quad (13.19)$$

for all  $S \in \mathbb{R}^{2 \times 2}$ . (Here  $V_1 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$  and  $V_2 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$ , thus  $W = [V_1 \ V_2]$ .) In particular,

$$2L_1(x, y) = V_1^*L_2(x, y)V_1 + V_2^*L_2(x, y)V_2. \quad (13.20)$$

Hence  $L_2(X_1, X_2) > 0$  implies  $L_1(X_1, X_2) > 0$ , i.e.,  $\mathcal{D}_{L_2} \subseteq \mathcal{D}_{L_1}$ .

The computations leading up to (13.20) illustrate the proof of our linear Positivstellensatz, Theorem 13.1. For the details see [18, Sect. 3.1].

### 13.2.6 Positivstellensatz on a Spectrahedron

Our non-commutative techniques lead to a cleaner and more powerful commutative Putinar-type Positivstellensatz [45] for  $p$  strictly positive on a bounded spectrahedron  $\overline{\mathcal{D}}_L(1) = \{x \in \mathbb{R}^g : L(x) \geq 0\}$ . In the theorem which follows,  $\mathbb{SR}^{d \times d}[y]$  is the set of symmetric  $d \times d$  matrices with entries from  $\mathbb{R}[y]$ , the algebra of (commutative) polynomials with coefficients from  $\mathbb{R}$ . Note that an element of  $\mathbb{SR}^{d \times d}[y]$  may be identified with a polynomial (in commuting variables) with coefficients from  $\mathbb{SR}^{d \times d}$ .

**Theorem 13.4.** Suppose  $L \in \mathbb{SR}^{d \times d}[y]$  is a monic linear pencil and  $\overline{\mathcal{D}}_L(1)$  is bounded. Then for every symmetric matrix polynomial  $p \in \mathbb{R}^{\ell \times \ell}[y]$  with  $p|_{\overline{\mathcal{D}}_L(1)} > 0$ , there are  $A_j \in \mathbb{R}^{\ell \times \ell}[y]$ , and  $B_k \in \mathbb{R}^{d \times \ell}[y]$  satisfying

$$p = \sum_j A_j^* A_j + \sum_k B_k^* L B_k. \quad (13.21)$$

The Positivstellensatz, Theorem 13.4, has a non-commutative version for  $\delta \times \delta$  matrix valued symmetric polynomials  $p$  in non-commuting variables positive on a nc LMI set  $\overline{\mathcal{D}}_L$ , see [18]. In the case this matrix valued polynomial  $p$  is linear, this Positivstellensatz reduces to Theorem 13.1, which can thus be regarded as a “Linear Positivstellensatz”. For perspective we mention that the proofs of our Positivstellensätze actually rely on the linear Positivstellensatz. For experts we point out that the key reason LMI sets behave better is that the quadratic module associated to a monic linear pencil  $L$  with bounded  $\overline{\mathcal{D}}_L$  is archimedean.

We shall return to the topic of Positivstellensätze in Sect. 13.5.

### 13.3 Non-commutative Convex Semi-algebraic Sets Are LMI Representable

The main result of this section is that a bounded convex non-commutative basic open semi-algebraic set has a monic Linear Matrix Inequality representation. Applications and connections to semidefinite programming and linear systems engineering are discussed in Sect. 13.3.4. The work is also of interest in understanding a non-commutative (free) analog of convex semi-algebraic sets [4].

For perspective, in the commutative case of a basic open semi-algebraic subset  $C$  of  $\mathbb{R}^g$ , there is a stringent condition, called the “line test”, which, in addition to convexity, is necessary for  $C$  to have an LMI representation. In two dimensions the line test is necessary and sufficient, [29], a result used by Lewis–Parrilo–Ramana [35] to settle a 1958 conjecture of Peter Lax on hyperbolic polynomials. Indeed LMI representations are closely tied to properties of hyperbolic polynomials; see this volume, the survey of Helton and Nie.

In summary, if a (commutative) bounded basic open semi-algebraic convex set has an LMI representation, then it must pass the highly restrictive line test; whereas a non-commutative bounded basic open semi-algebraic set has an LMI representation if and only if it is convex.

A subset  $\mathcal{S}$  of  $(\mathbb{SR}^{n \times n})^g$  is closed under unitary conjugation if for every  $X = (X_1, \dots, X_g) \in \mathcal{S}$  and  $U$  a  $n \times n$  unitary, we have  $U^* X U = (U^* X_1 U, \dots, U^* X_g U) \in \mathcal{S}$ . The sequence  $C = (C(n))_{n \in \mathbb{N}}$ , where  $C(n) \subseteq (\mathbb{SR}^{n \times n})^g$ , is a **non-commutative set** if it is closed under unitary conjugation and direct sums; i.e., if  $X = (X_1, \dots, X_g) \in C(n)$  and  $Y = (Y_1, \dots, Y_g) \in C(m)$ , then  $X \oplus Y = (X_1 \oplus Y_1, \dots, X_g \oplus Y_g) \in C(n+m)$ . Such set  $C$  has an **LMI representation** if there is a monic linear pencil  $L$  such that

$$C = \mathcal{D}_L.$$

Of course, if  $C = \mathcal{D}_L$ , then the closure  $\overline{C}$  of  $C$  has the representation  $\{X : L(X) \geq 0\}$  and so we could also refer to  $\overline{C}$  as having an LMI representation.

Clearly, if  $C$  has an LMI representation, then  $C$  is a convex non-commutative basic open semi-algebraic set. The main result of this section is the converse, under the additional assumption that  $C$  is bounded.

Since we are dealing with matrix convex sets, it is not surprising that the starting point for our analysis is the matricial version of the Hahn–Banach Separation Theorem of Effros and Winkler [13] which says that given a point  $x$  not inside a matrix convex set there is a (finite) LMI which separates  $x$  from the set. For a general matrix convex set  $C$ , the conclusion is then that there is a collection, likely infinite, of finite LMIs which cut out  $C$ .

In the case  $C$  is matrix convex and also semi-algebraic, the challenge is to prove that there is actually a finite collection of (finite) LMIs which define  $C$ . The techniques used to meet this challenge have little relation to previous work on convex non-commutative basic semi-algebraic sets. In particular, they do not involve non-commutative calculus and positivity. See [21] for the details.

### 13.3.1 Non-commutative Basic Open Semi-algebraic Sets

Suppose  $p \in \mathbb{R}\langle x \rangle^{\delta \times \delta}$  is symmetric. In particular,  $p(0)$  is a  $\delta \times \delta$  symmetric matrix. Assume that  $p(0) > 0$ . For each positive integer  $n$ , let

$$\mathfrak{I}_p(n) = \{X \in (\mathbb{SR}^{n \times n})^g : p(X) > 0\},$$

and define  $\mathfrak{I}_p$  to be the sequence (graded set)  $(\mathfrak{I}_p(n))_{n=1}^\infty$ . Let  $\mathcal{D}_p(n)$  denote the connected component of 0 of  $\mathfrak{I}_p(n)$  and  $\mathcal{D}_p$  the sequence (graded set)  $(\mathcal{D}_p(n))_{n=1}^\infty$ . We call  $\mathcal{D}_p$  the **positivity set** of  $p$ . In analogy with classical real algebraic geometry we call sets of the form  $\mathcal{D}_p$  **non-commutative basic open semi-algebraic sets**.

*Remark 13.1.* By a simple affine linear change of variable the point 0 can be replaced by  $\lambda \in \mathbb{R}^q$ . Replacing 0 by a fixed  $\Lambda \in (\mathbb{SR}^{n \times n})^g$  would require an extension of the theory.

### 13.3.2 Convex Semi-algebraic Sets

To say that  $\mathcal{D}_p$  is **convex** means that each  $\mathcal{D}_p(n)$  is convex (in the usual sense) and in this case we say  $\mathcal{D}_p$  is a **convex non-commutative basic open semi-algebraic set**. In addition, we generally assume that  $\mathcal{D}_p$  is bounded; i.e., there is a constant  $K$  such for each  $n$  and each  $X \in \mathcal{D}_p(n)$ , we have  $\|X\| = \sum \|X_j\| \leq K$ . Thus the following list of conditions summarizes our usual assumptions on  $p$ .

**Assumption 13.1** Fix  $p$  a  $\delta \times \delta$  symmetric matrix of polynomials in  $g$  non-commuting variables of degree  $d$ . Our standard assumptions are:

- (1)  $p(0)$  is positive definite;
- (2)  $\mathcal{D}_p$  is bounded; and
- (3)  $\mathcal{D}_p$  is convex.

### 13.3.3 The Result

Our main theorem of this section is

**Theorem 13.5 ([21]).** Every convex non-commutative bounded basic open semi-algebraic set (as in Assumption 13.1) has an LMI representation.

The proof of Theorem 13.5 yields estimates on the size of the representing LMI.

**Theorem 13.6.** Suppose  $p$  satisfies the conditions of Assumption 13.1. Thus  $p$  is a symmetric  $\delta \times \delta$ -matrix polynomial of degree  $d$  in  $g$  variables. Let  $v = \delta \sum_{j=0}^d g^j$ .

Then the size of the matrices in  $L$  is no greater than  $\frac{\check{v}(\check{v}+1)}{2}$ , where  $\check{v} = \delta \sum_{j=0}^{\lceil \frac{d}{2} \rceil} g^j$ .

As usual,  $\lceil \frac{d}{2} \rceil$  stands for the smallest integer  $\geq \frac{d}{2}$ . Of course

$$\left\lceil \frac{d}{2} \right\rceil = \frac{d}{2} \text{ when } d \text{ is even} \quad \text{and} \quad \left\lceil \frac{d}{2} \right\rceil = \frac{d+1}{2} \text{ when } d \text{ is odd.}$$

The results above hold even if sets more general than  $\mathcal{D}_p$  are used. Suppose  $p(0)$  is invertible and define  $\mathcal{I}_p$  to be the component of  $\{X : p(X) \text{ is invertible}\}$  containing 0. Then if  $\mathcal{I}_p$  is bounded and convex, the theorems above still hold for  $\mathcal{I}_p$ ; it has an LMI representation.

An unexpected consequence of Theorem 13.5 is that projections of non-commutative semi-algebraic sets may not be semi-algebraic. For details and proofs see [21].

### 13.3.4 Motivation

One of the main advances in systems engineering in the 1990s was the conversion of a set of problems to LMIs, since LMIs, up to modest size, can be solved numerically by semidefinite programs [48]. A large class of linear systems problems are described in terms of a signal-flow diagram  $\Sigma$  plus  $L^2$  constraints (such as energy dissipation). Routine methods convert such problems into a non-commutative polynomial inequality of the form  $p(X) \geq 0$  or  $p(X) > 0$ .

Instantiating specific systems of linear differential equations for the “boxes” in the system flow diagram amounts to substituting their coefficient matrices for variables in the polynomial  $p$ . Any property asserted to be true must hold when matrices of any size are substituted into  $p$ . Such problems are referred to as dimension free. We emphasize, the polynomial  $p$  itself is determined by the signal-flow diagram  $\Sigma$ .

Engineers vigorously seek convexity, since optima are global and convexity lends itself to numerics. Indeed, there are over a thousand papers trying to convert linear systems problems to convex ones and the only known technique is the rather blunt trial and error instrument of trying to guess an LMI. Since having an LMI is seemingly more restrictive than convexity, there has been the hope, indeed expectation, that some practical class of convex situations has been missed. The problem solved here (though not operating at full engineering generality, see [14]) is a paradigm for the type of algebra occurring in systems problems governed by signal-flow diagrams; such physical problems directly present non-commutative semi-algebraic sets. Theorem 13.5 gives compelling evidence that all such convex

situations are associated to some LMI. Thus we think the implications of our results here are negative for linear systems engineering; for dimension free problems there is no convexity beyond LMIs.

A basic question regarding the range of applicability of SDP is: which sets have an LMI representation? Theorem 13.5 settles, to a reasonable extent, the case where the variables are non-commutative (effectively dimension free matrices).

### 13.4 Convex Polynomials

We turn now from non-commutative convex sets to non-commutative convex polynomials. If  $p$  is concave ( $-p$  is convex) and monic, then the set  $S = \{X : p(X) > 0\}$  is a convex non-commutative basic open semialgebraic. If it is also bounded, then, by the results of the previous section, it has an LMI representation. However, much more is true and the analysis turns on a nc version of the Hessian and connects with nc semi-algebraic geometry.

A symmetric polynomial  $p$  is **matrix convex**, or simply **convex** for short, if for each positive integer  $n$ , each pair of tuples  $X \in (\mathbb{SR}^{n \times n})^g$  and  $Y \in (\mathbb{SR}^{n \times n})^g$ , and each  $0 \leq t \leq 1$ ,

$$p(tX + (1-t)Y) \leq tp(X) + (1-t)p(Y). \quad (13.22)$$

Even in one-variable, convexity in the non-commutative setting differs from convexity in the commuting case because here  $Y$  need not commute with  $X$ . For example, to see that the polynomial  $p = x^4$  is not matrix convex, let

$$X = \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \text{ and } Y = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$$

and compute

$$\frac{1}{2}X^4 + \frac{1}{2}Y^4 - \left(\frac{1}{2}X + \frac{1}{2}Y\right)^4 = \begin{bmatrix} 164 & 120 \\ 120 & 84 \end{bmatrix}$$

which is not positive semidefinite. On the other hand, to verify that  $x^2$  is a matrix convex polynomial, observe that

$$tX^2 + (1-t)Y^2 - (tX + (1-t)Y)^2 = t(1-t)(X - Y)^2 \geq 0.$$

It is possible to automate checking for convexity, rather than depending upon lucky choices of  $X$  and  $Y$  as was done above. The theory described in [10], leads to and validates a symbolic algorithm for determining regions of convexity of non-commutative polynomials and even of non-commutative rational functions (for non-commutative rationals see [27, 33]) which is implemented in NCAlgebra.

Let us illustrate it on the example  $p(x) = x^4$ . The NCAlgebra command is  
**NCConvexityRegion[Function F, {Variables x}].**

```
In[1]:= SetNonCommutative[x];
In[2]:= NCConvexityRegion[ x***x***x***x, {x} ]
Out[2]:= { {2, 0, 0}, {0, 2}, {0, -2} }
```

which we interpret as saying that  $p(x) = x^4$  is convex on the set of matrices  $X$  for which the the  $3 \times 3$  block matrix valued non-commutative function

$$\rho(X) = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 0 & 2 \\ 0 & -2 & 0 \end{bmatrix} \quad (13.23)$$

is positive semidefinite. Since  $\rho(X)$  is constant and never positive semidefinite, we conclude that  $p$  is *nowhere* convex.

This example is a simple special case of the following theorem.

**Theorem 13.7 ([22]).** *Every convex symmetric polynomial in the free algebra  $\mathbb{R}\langle x \rangle$  has degree two or less.*

### 13.4.1 The Proof of Theorem 13.7 and Its Ingredients

Just as in the commutative case, convexity of a symmetric  $p \in \mathbb{R}\langle x \rangle$  is equivalent to positivity of its Hessian  $q(x)[h]$  which is a polynomial in the  $2g$  variables  $x = (x_1, \dots, x_g)$  and  $h = (h_1, \dots, h_g)$ . Unlike the commutative case, a positive non-commutative polynomial is a sum of squares. Thus, if  $p$  is convex, then its Hessian  $q(x)[h]$  is a sum of squares. Combinatorial considerations say that a Hessian which is also a sum of squares must come from a polynomial of degree two. In the remainder of this section we flesh out this argument, introducing the needed definitions, techniques, and results.

#### 13.4.1.1 Non-commutative Derivatives

For practical purposes, the  $k$ th-**directional derivative** of a nc polynomial  $p$  is given by

$$p^{(k)}(x)[h] = \frac{d^k}{dt^k} p(x + th) \Big|_{t=0}.$$

Note that  $p^{(k)}(x)[h]$  is homogeneous of degree  $k$  in  $h$  and moreover, if  $p$  is symmetric, then so is  $p^k(x)[h]$ . For  $X, H \in (\mathbb{SR}^{n \times n})^g$  observe that

$$p'(X)[H] = \lim_{t \rightarrow 0} \frac{p(X + tH) - p(X)}{t}.$$

*Example 13.1.* The one variable polynomial  $p(x) = x^4$  has first derivative

$$p'(x)[h] = hxxx + xhxx + xxhx + xxxh.$$

Note each term is linear in  $h$  and  $h$  replaces each occurrence of  $x$  once and only once. The Hessian, or second derivative, of  $p$  is

$$p''(x)[h] = 2hhxx + 2hxhx + 2hxxh + 2xhhx + 2xhxx + 2xxhh.$$

Note each term is degree two in  $h$  and  $h$  replaces each pair of  $x$ 's exactly once.

**Theorem 13.8 ([28]).** *Every symmetric polynomial  $p \in \mathbb{R}\langle x \rangle$  whose  $k$ th derivative is a matrix positive polynomial has degree  $k$  or less.*

*Proof.* See [28] for the full proof or [22] for case of  $k = 2$ . The very intuitive proof based upon a little non-commutative semi-algebraic geometry is sketched in the next subsection.  $\square$

### 13.4.1.2 A Little Non-commutative Semi-algebraic Geometry

The proof of Theorem 13.7 employs the most fundamental of all non-commutative Positivstellensätze.

A symmetric non-commutative polynomial  $p$  is **matrix positive** or simply **positive** provided  $p(X_1, \dots, X_g)$  is positive semidefinite for every  $X \in (\mathbb{SR}^{n \times n})^g$  (and every  $n$ ). An example of a matrix positive polynomial is a **Sum of Squares** of polynomials, meaning an expression of the form

$$p(x) = \sum_{j=1}^c h_j(x)^* h_j(x).$$

Substituting  $X \in (\mathbb{SR}^{n \times n})^g$  gives  $p(X) = \sum_{j=1}^c h_j(X)^* h_j(X) \geq 0$ . Thus  $p$  is positive. Remarkably these are the only positive non-commutative polynomials.

**Theorem 13.9 ([15]).** *Every matrix positive polynomial is a sum of squares.*

This theorem is just a sample of the structure of non-commutative semi-algebraic geometry, the topic of Sect. 13.5.

Suppose  $p \in \mathbb{R}\langle x \rangle$  is (symmetric and) convex and  $Z, H \in (\mathbb{SR}^{n \times n})^g$  and  $t \in \mathbb{R}$  are given. In the definition of convex, choosing  $X = Z + tH$  and  $Y = Z - tH$ , it follows that

$$0 \leq p(Z + tH) + p(Z - tH) - 2p(Z),$$

and therefore

$$0 \leq \lim_{t \rightarrow 0} \frac{p(X + tH) + p(X - tH) - 2p(X)}{t^2} = p''(X)[H].$$

Thus the Hessian of  $p$  is matrix positive and since, in the non-commutative setting, positive polynomials are sums of squares we obtain the following theorem.

**Proposition 13.1.** *If  $p$  is matrix convex, then its Hessian  $p''(x)[h]$  is a sum of squares.*

#### 13.4.1.3 Proof of Theorem 13.7 by Example

Here we illustrate the proof of Theorem 13.7 based upon Proposition 13.1 by showing that  $p(x) = x^4$  is not matrix convex. Indeed, if  $p(x)$  is matrix convex, then  $p''(x)[h]$  is matrix positive and therefore, by Proposition 13.1, there exists a  $\ell$  and polynomials  $f_1(x, h), \dots, f_\ell(x, h)$  such that

$$\begin{aligned} \frac{1}{2}p''(x)[h] &= hhxx + hxhx + hxxh + xhhx + xhxh + xxhh \\ &= f_1(x, h)^* f_1(x, h) + \dots + f_\ell(x, h)^* f_\ell(x, h). \end{aligned}$$

One can show that each  $f_j(x, h)$  is linear in  $h$ . On the other hand, some term  $f_i^* f_i$  contains  $hhxx$  and thus  $f_i$  contains  $hx^2$ . Let  $m$  denote the largest  $\ell$  such that some  $f_j$  contains the term  $hx^\ell$ . Then  $m \geq 1$  and for such  $j$ , the product  $f_j^* f_j$  contains the term  $hx^{2m}h$  which cannot be cancelled out, a contradiction.  $\square$

The proof of the more general, order  $k$  derivative, is similar, see [28].

#### 13.4.2 Non-commutative Rational and Analytic Functions

A class of functions bigger than nc polynomials is given by nc analytic functions, see e.g. Voiculescu [49, 50] or the forthcoming paper of Kaliuzhnyi–Verbovetskyi and Vinnikov for an introduction. The rigidity of nc bianalytic maps is investigated by Popescu [44]; see also [17, 19, 20]. For other properties of nc analytic functions, a very interesting body of work, e.g. by Popescu [43] can be used as a gateway.

The articles [5, 27, 33] deal with non-commutative rational functions. For instance, [27] shows that if a non-commutative rational function is convex in an open set, then it is the Schur Complement of some monic linear pencil.

## 13.5 Algebraic Certificates of Positivity

In this section we give a brief overview of various free  $*$ -algebra analogs to the classical Positivstellensätze, i.e., theorems characterizing polynomial inequalities in a purely algebraic way. Here it is of benefit to consider free *non-symmetric* variables. That is, let  $x = (x_1, \dots, x_g)$  be non-commuting variables and  $x^* = (x_1^*, \dots, x_g^*)$  another set of non-commuting variables. Then  $\mathbb{R}\langle x, x^* \rangle$  is the free  $*$ -algebra of polynomials in the non-commuting indeterminates  $x, x^*$ .

There is a natural involution  $*$  on  $\mathbb{R}\langle x, x^* \rangle$  induced by  $x_i \mapsto x_i^*$  and  $x_j^* \mapsto x_j$ . As before,  $p \in \mathbb{R}\langle x, x^* \rangle$  is symmetric if  $p = p^*$ . An element of the form  $p^* p$  is a square, and  $\Sigma^2$  denotes the convex cone of all sums of squares. Given a matrix polynomial  $p = \sum_w p_w w \in \mathbb{R}\langle x, x^* \rangle^{\delta \times \delta'}$  and  $X \in (\mathbb{R}^{n \times n})^g$ , we define the evaluation  $p(X, X^*)$  by analogy with evaluation in the symmetric variable case.

### 13.5.1 Positivstellensätze

This subsection gives an indication of various free  $*$ -algebra analogs to the classical theorems characterizing polynomial inequalities in a purely algebraic way. We will start by sketching a proof of the following refinement of Theorem 13.9.

**Theorem 13.10 ([15]).** *Let  $p \in \mathbb{R}\langle x, x^* \rangle_d$  be a non-commutative polynomial. If  $p(M, M^*) \geq 0$  for all  $g$ -tuples of linear operators  $M$  acting on a Hilbert space of dimension at most  $N(k) := \dim \mathbb{R}\langle x, x^* \rangle_k$  with  $2k \geq d + 2$ , then  $p \in \Sigma^2$ .*

*Proof.* Note that a polynomial  $p$  satisfying the hypothesis automatically satisfies  $p = p^*$ . The only necessary technical result we need is the closedness of the cone  $\Sigma_k^2$  in the Euclidean topology of the finite dimensional space  $\mathbb{R}\langle x, x^* \rangle_k$ . This is done as in the commutative case, using Carathéodory's convex hull theorem. More exactly, every polynomial of  $\Sigma_k^2$  is a convex combination of at most  $\dim \mathbb{R}\langle x, x^* \rangle_k + 1$  squares (of polynomials). On the other hand the positive functionals on  $\Sigma_k^2$  separate the points of  $\mathbb{R}\langle x, x^* \rangle_k$ . See for details [24].

Assume that  $p \notin \Sigma^2$  and let  $k \geq (d+2)/2$ , so that  $p \in \mathbb{R}\langle x, x^* \rangle_{2k-2}$ . Once we know that  $\Sigma_{2k}^2$  is a closed cone, we can invoke Minkowski separation theorem and find a symmetric functional  $L \in \mathbb{R}\langle x, x^* \rangle'_{2k}$  providing the strict separation:

$$L(p) < 0 \leq L(f), \quad f \in \Sigma_{2k}^2.$$

Applying the Gelfand–Naimark–Segal construction to  $L$  yields a tuple  $M$  of operators acting on a Hilbert space  $H$  of dimension  $N(k)$  and a vector  $\xi \in H$ , such that

$$0 \leq \langle p(M, M^*)\xi, \xi \rangle = L(p) < 0,$$

a contradiction.  $\square$

When compared to the commutative framework, this theorem is stronger in the sense that it does not assume a strict positivity of  $p$  on a well chosen “spectrum”. Variants with supports (for instance for spherical tuples  $M : M_1^*M_1 + \dots + M_g^*M_g \leq I$ ) of the above result are discussed in [24].

To draw a very general conclusion from the above computations: when dealing with positivity in a free  $*$ -algebra, the standard point evaluations (or more precisely prime or real spectrum evaluations) of the commutative case are replaced by matrix evaluations of the free variables. The positivity can be tailored to “evaluations in a supporting set”. The results pertaining to the resulting algebraic decompositions are called Positivstellensätze, see [40] for details in the commutative setting. We state below an illustrative and generic result, from [23], for sums of squares decompositions in a free  $*$ -algebra.

**Theorem 13.11 ([23]).** *Let  $p = p^* \in \mathbb{R}\langle x, x^* \rangle$  and let  $q = \{q_1, \dots, q_k\} \subseteq \mathbb{R}\langle x, x^* \rangle$  be a set of symmetric polynomials, so that*

$$\text{QM}(q) = \text{co}\{f^* q_i f; f \in \mathbb{R}\langle x, x^* \rangle, 0 \leq i \leq k\}, \quad q_0 = 1,$$

*contains  $1 - x_1^*x_1 - \dots - x_g^*x_g$ . If for all tuples of linear bounded Hilbert space operators  $X = (X_1, \dots, X_g)$ , we have*

$$q_i(X, X^*) \geq 0, \quad 1 \leq i \leq k \quad \Rightarrow \quad p(X, X^*) > 0, \quad (13.24)$$

*then  $p \in \text{QM}(q)$ .*

Henceforth, call  $\text{QM}(q)$  the **quadratic module** generated by the set of polynomials  $q$ .

We omit the proof of Theorem 13.11, as it is very similar to the previous proof. The only difference is in the separation theorem applied. For details, see [23].

Some interpretation is needed in degenerate cases, such as those where no bounded operators satisfy the relations  $q_i(X, X^*) \geq 0$ . Suppose for example, if  $\phi$  denotes the defining relations for the Weyl algebra and the  $q_i$  include  $-\phi^*\phi$ . In this case, we would say  $p(X, X^*) > 0$ , since there are no  $X$  satisfying  $q(X, X^*)$ , and voila  $p \in \text{QM}(q)$  as the theorem says. A non-archimedean Positivstellensatz for the Weyl algebra, which treats unbounded representations and eigenvalues of polynomial partial differential operators, is given in [46].

A paradigm practical question with matrix inequalities is:

*Given a non-commutative symmetric polynomial  $p(a, x)$  and a  $n \times n$  matrix tuple  $A$ , find  $X \geq 0$ , if possible, which makes  $p(A, X) \geq 0$ .*

As a refinement of this problem, let  $q(a, x)$  be a given nc symmetric polynomial. For a given  $A$ , find  $X$  if possible, such that both  $q(A, X)$  and  $p(A, X)$  are positive semidefinite. The infeasibility of this latter problem is equivalent to the statement, if  $q(A, X) \geq 0$ , then  $p(A, X) \not\geq 0$ . There is keen interest in numerical solutions of such problems. The next theorem informs us that the main issue is the matrix coefficients  $A$ , as it gives a “certificate of infeasibility” for the problem in the absense of  $A$ .

**Theorem 13.12 (Nirgendsnegativsemidefinitsstellensatz [30]).**

Let  $p = p^* \in \mathbb{R}\langle x, x^* \rangle$  and let  $q = \{q_1, \dots, q_k\} \subset \mathbb{R}\langle x, x^* \rangle$  be a set of symmetric polynomials so that  $\text{QM}(q)$  contains  $1 - x_1^* x_1 - \dots - x_g^* x_g$ . If for all tuples of linear bounded Hilbert space operators  $X = (X_1, \dots, X_g)$ , we have

$$q_i(X, X^*) \geq 0, \quad 1 \leq i \leq k \quad \Rightarrow \quad p(X, X^*) \not\leq 0, \quad (13.25)$$

then there exists an integer  $r$  and  $h_1, \dots, h_r \in \mathbb{R}\langle x, x^* \rangle$  with  $\sum_{i=1}^r h_i^* p h_i \in 1 + \text{QM}(q)$ .

*Proof.* By (13.25),

$$\{X \mid q_i(X, X^*) \geq 0, \quad 1 \leq i \leq k, \quad -p(X, X^*) \geq 0\} = \emptyset.$$

Hence  $-1 \in \text{QM}(q, -p)$  by Theorem 13.11.  $\square$

### 13.5.2 Quotient Algebras

The results from Sect. 13.5.1 allow a variety of specializations to quotient algebras. In this subsection we consider a two sided ideal  $\mathcal{I}$  of  $\mathbb{R}\langle x, x^* \rangle$  which need not be invariant under  $*$ . Then one can replace the quadratic module  $\text{QM}$  in the statement of a Positivstellensatz with  $\text{QM}(q) + \mathcal{I}$ , and apply similar arguments as above. For instance, the next simple observation can be deduced.

**Corollary 13.1.** *Assume, in the hypotheses of Theorem 13.11, that the relations (13.24) include some relations of the form  $r(X, X^*) = 0$ , even with  $r$  not symmetric, then*

$$p \in \text{QM}(q) + \mathcal{I}_r \quad (13.26)$$

where  $\mathcal{I}_r$  denotes the two sided ideal generated by  $r$ .

*Proof.* This follows immediately from  $p \in \text{QM}(q, -r^* r)$  which is a consequence of Theorem 13.11 and the fact

$$\text{QM}(q, -r^* r) \subset \text{QM}(q) + \mathcal{I}_r.$$

$\square$

For instance, we can look at the situation where  $r$  is the commutator  $[x_i, x_j]$  as insisting on positivity of  $q(X)$  only on commuting tuples of operators, in which case the ideal  $\mathcal{I}$  generated by  $[x_j^*, x_i^*]$ ,  $[x_i, x_j]$  is added to  $\text{QM}(q)$ . The classical commuting case is captured by the corollary applied to the “commutator ideal”:  $\mathcal{I}_{[x_j^*, x_i^*], [x_i, x_j], [x_i, x_j^*]}$  for  $i, j = 1, \dots, g$  which requires testing only on commuting tuples of operators drawn from a commuting  $C^*$ -algebra. The classical Spectral Theorem, then converts this to testing only on  $\mathbb{C}^g$ , cf. [28].

The situation where one tests for constrained positivity in the absence of an archimedean property is thoroughly analyzed in [47].

### 13.5.3 A Nullstellensatz

With similar techniques (well chosen, separating,  $*$ -representations of the free algebra) and a rather different “dilation type” of argument, one can prove a series of Nullstellensätze.

We state for information one of them. For an early version see [25].

**Theorem 13.13.** *Let  $q_1(x), \dots, q_m(x) \in \mathbb{R}\langle x \rangle$  be polynomials not depending on the  $x_j^*$  variables and let  $p(x, x^*) \in \mathbb{R}\langle x, x^* \rangle$ . Assume that for every  $g$  tuple  $X$  of linear operators acting on a finite dimensional Hilbert space  $H$ , and every vector  $v \in H$ , we have:*

$$(q_j(X)v = 0, 1 \leq j \leq m) \Rightarrow (p(X, X^*)v = 0).$$

*Then  $p$  belongs to the left ideal  $\mathbb{R}\langle x, x^* \rangle q_1 + \dots + \mathbb{R}\langle x, x^* \rangle q_m$ .*

Again, this proposition is stronger than its commutative counterpart. For instance there is no need of taking higher powers of  $p$ , or of adding a sum of squares to  $p$ . Note that here  $\mathbb{R}\langle x \rangle$  has a different meaning than earlier, since, unlike previously, the variables are nonsymmetric.

We refer the reader to [26] for the proof of Theorem 13.13. An earlier, transpose-free Nullstellensatz due to Bergman was given in [23].

Here is a theorem which could be regarded as a very different type of non-commutative Nullstellensatz.

**Theorem 13.14 ([31]).** *Let  $p = p^* \in \mathbb{R}\langle x, x^* \rangle_d$  be a non-commutative polynomial satisfying  $\text{tr } p(M, M^*) = 0$  for all  $g$ -tuples of linear operators  $M$  acting on a Hilbert space of dimension at most  $d$ . Then  $p$  is a sum of commutators of non-commutative polynomials.*

We end this subsection with an example which goes against any intuition we would carry from the commutative case, see [23].

*Example 13.2.* Let  $q = (x^* x + x x^*)^2$  and  $p = x + x^*$  where  $x$  is a single variable. Then, for every matrix  $X$  and vector  $v$  (belonging to the space where  $X$  acts),  $q(X)v = 0$  implies  $p(X)v = 0$ ; however, there does not exist a positive integer  $m$  and  $r, r_j \in \mathbb{R}\langle x, x^* \rangle$ , so that

$$p^{2m} + \sum r_j^* r_j = qr + r^* q. \quad (13.27)$$

Moreover, we can modify the example to add the condition  $q(X)$  is positive semidefinite implies  $p(X)$  is positive semidefinite and still not obtain this representation.

### 13.5.4 Tracial Positivstellensatz

Another type of non-commutative positivity is given by the trace. A polynomial  $p \in \mathbb{R}\langle x, x^* \rangle$  is called **trace-positive** if  $\text{tr } p(X, X^*) \geq 0$  for all  $X \in (\mathbb{R}^{n \times n})^g$ . The main motivation for studying these comes from two outstanding open problems: Connes' embedding conjecture [12] from operator algebras [31] and the Bessis–Moussa–Villani (BMV) conjecture [8] from quantum statistical mechanics [32].

Clearly, a sum of a matrix positive (i.e., sum of hermitian squares by Theorem 13.10) and a trace-zero (i.e., sum of commutators by Theorem 13.14) polynomial is trace-positive. However, unlike in the matrix positive case, not every trace-positive polynomial is of this form [31, 32].

*Example 13.3.* Let  $x$  denote a single non-symmetric variable and

$$\begin{aligned} M_0 := & 3x^4 - 3(xx^*)^2 - 4x^5x^* - 2x^3(x^*)^3 \\ & + 2x^2x^*x(x^*)^2 + 2x^2(x^*)^2xx^* + 2(xx^*)^3. \end{aligned}$$

Then the non-commutative Motzkin polynomial in non-symmetric variables is

$$M := 1 + M_0 + M_0^* \in \mathbb{R}\langle x, x^* \rangle.$$

It is trace-positive but is not a sum of hermitian squares and commutators.

Life is somewhat easier in the constrained, bounded case. For instance, in the language of operator algebras we have:

**Theorem 13.15 ([31]).** *For  $f = f^* \in \mathbb{R}\langle x, x^* \rangle$  the following are equivalent:*

- (i)  $\text{tr}(f(a, a^*)) \geq 0$  for all finite von Neumann algebras  $\mathcal{A}$  and all tuples of contractions  $a \in \mathcal{A}^g$ ;
- (ii) For every  $\varepsilon \in \mathbb{R}_{>0}$ ,  $f + \varepsilon$  is a sum of commutators and of an element from  $\text{QM}(1 - x_1^*x_1, \dots, 1 - x_g^*x_g)$ .

The big open question [12, 31] is whether (i) or (ii) is equivalent to

- (iii)  $\text{tr}(f(X, X^*)) \geq 0$  for all  $n \in \mathbb{N}$  and all tuples of contractions  $X \in (\mathbb{R}^{n \times n})^g$ .

An attempt at better understanding trace-positivity is made in [6], where the duality between trace-positive polynomials and the *tracial moment problem* is exploited. The tracial moment problem is the following question: For which sequences  $(y_w)$  indexed by words  $w$  in  $x, x^*$ , does there exist  $n \in \mathbb{N}$ , and a positive Borel measure  $\mu$  on  $(\mathbb{SR}^{n \times n})^n$  satisfying

$$y_w = \int w(A) d\mu(A) ? \tag{13.28}$$

Such a sequence is a *tracial moment sequence*. If one is interested only in *finite* sequences  $(y_w)$ , then this is the *truncated* tracial moment problem.

To a sequence  $y = (y_w)$  one associates the (infinite) *Hankel matrix*  $M(y)$ , indexed by words, by  $M(y)_{u,v} = y_{u^*v}$ . One of the results in [6] shows that if  $M(y)$  is positive semidefinite and of finite rank, then  $y$  is a tracial moment sequence. In the truncated case a condition called “flatness” governs the existence of a representing measure, much like in the classical case. For details and proofs see [6].

For the free non-commutative moment problem we refer the reader to [36].

## 13.6 Algebraic Software

This section briefly surveys existing software dealing with non-commutative convexity (Sect. 13.6.1) and positivity (Sect. 13.6.2).

### 13.6.1 NCAlgebra Under Mathematica

Here is a list of software running under NCAlgebra [16] (which runs under Mathematica) that implements and experiments on symbolic algorithms pertaining to non-commutative Convexity and LMIs.

NCAlgebra is available from <http://www.math.ucsd.edu/~ncalg>

- **Convexity Checker.** Camino et al. [10] have an (algebraic) algorithm for determining the region on which a rational expression is convex.
- **Classical Production of LMIs.** There are two Mathematica NCAlgebra notebooks by de Oliveira and Helton. The first is based on algorithms for implementing the 1997 approach of Skelton et al. [48] associating LMIs to more than a dozen control problems. The second (requires C++ and NCGB) produces LMIs by symbolically implementing the 1997 change of variables method of Scherer et al.
- **Schur Complement Representations of a non-commutative rational.** This computes a linear pencil whose Schur complement is the given nc rational function  $p$  using Shopple – Slinglend thesis algorithm. It is not known if  $p$  convex near 0 always leads to a monic pencil via this algorithm, but we never saw a counter example.
- **Determinantal Representations.** Finds Determinantal Representations of a given polynomial  $p$ . Shopple – Slinglend implement Slinglend’s thesis algorithm plus the [27] algorithm. Requires NCAlgebra.

See <http://www.math.ucsd.edu/~ncalg/surveydemo> for occurrences of available demos.

### 13.6.2 NCSOStools Under Matlab

NCSOStools [11] which runs under Matlab, implements and experiments on numeric algorithms pertaining to non-commutative positivity and sums of squares. Here is a sample of features available.

- **Non-commuting variables.** Basic symbolic computation with nc variables for Matlab has been implemented.
- **Matrix-positivity.** An nc polynomial  $p$  is matrix positive if and only if it is a sum of squares. This can be easily tested using a variant of the classical Gram matrix method. Indeed,  $p \in \mathbb{R}\langle x \rangle_{2d}$  is a sum of squares if and only if  $p = \langle x \rangle_d^* G \langle x \rangle_d$  for a positive semidefinite  $G$ . (Here,  $\langle x \rangle_d$  denotes a (column) vector of all words in  $x$  of degree  $\leq d$ .) This can be easily formulated as a feasibility semidefinite program (SDP).
- **Eigenvalue optimization.** Again, using SDP we can compute the smallest eigenvalue  $f^*$  a symmetric  $f \in \mathbb{R}\langle x \rangle$  can attain. That is,

$$f^* = \inf\{\langle f(A)v, v \rangle : A \text{ a } g\text{-tuple of symmetric matrices, } v \text{ a unit vector}\}.$$

Hence  $f^*$  is the greatest lower bound on the eigenvalues  $f(A)$  can attain for  $g$ -tuples of symmetric matrices  $A$ , i.e.,  $(f - f^*)(A) \geq 0$  for all  $n$ -tuples of symmetric matrices  $A$ , and  $f^*$  is the largest real number with this property. Given that a polynomial is matrix positive if and only if it is a sum of squares we can compute  $f^*$  efficiently with SDP:

$$\begin{aligned} f^* &= \sup \lambda \\ \text{s. t. } & f - \lambda \in \Sigma^2. \end{aligned}$$

- **Minimizer extraction.** Unlike in the commutative case, if  $f^*$  is attained, then minimizers  $(A, v)$  can always be computed. That is,  $A$  is a  $g$ -tuple of symmetric matrices and  $v$  is a unit eigenvector for  $f(A)$  satisfying

$$f^* = \langle f(A)v, v \rangle.$$

Of course, in general  $f$  will not be bounded from below. Another problem is that even if  $f$  is bounded, the infimum  $f^*$  need not be attained. The core ingredient of this minimizer extraction is the nc moment problem governed by a condition called “flatness”, together with the GNS construction.

- **Commutators and Cyclic equivalence.** Two polynomials are cyclically equivalent if their difference is a sum of commutators. This is easy to check.
- **Trace-positivity.** The sufficient condition for trace-positivity (i.e., sum of squares up to cyclic equivalence) is tested for using a variant of the Gram matrix method applied to matrix positivity.

NCSOStools is extensively documented and available at <http://ncsostools.fis.unm.si>

## References

1. Arveson, W.: Subalgebras of  $C^*$ -algebras. *Acta Mathematica* **123**, 141–224 (1969)
2. Arveson, W.: Subalgebras of  $C^*$ -algebras. II. *Acta Mathematica* **128**(3-4), 271–308 (1972)
3. Arveson, W.: The noncommutative Choquet boundary. *Journal of the American Mathematical Society* **21**(4), 1065–1084 (2008)
4. Bochnack, J., Coste, M., Roy, M.-F.: Real algebraic geometry. *Ergebnisse der Mathematik und ihrer Grenzgebiete. 3. Folge A Series of Modern Surveys in Mathematics*, Vol. 36. Springer-Verlag, Berlin (1998)
5. Ball, J.A., Groenewald, G., Malakorn, T.: Bounded real lemma for structured noncommutative multidimensional linear systems and robust control. *Multidimensional Systems And Signal Processing* **17**(2-3), 119–150 (2006)
6. Burgdorf, S., Klep, I.: The truncated tracial moment problem. To appear in the *Journal of Operator Theory*, <http://arxiv.org/abs/1001.3679>
7. Blecher, D.P., Le Merdy, C.: Operator algebras and their modules—an operator space approach, *London Mathematical Society Monographs*, vol. 30. The Clarendon Press Oxford University Press, Oxford (2004)
8. Bessis, D., Moussa, P., Villani, M.: Monotonic converging variational approximations to the functional integrals in quantum statistical mechanics. *Journal of Mathematical Physics* **16**(11), 2318–2325 (1975)
9. Ben-Tal A., Nemirovski, A.: On tractable approximations of uncertain linear matrix inequalities affected by interval uncertainty. *SIAM Journal on Optimization* **12**(3), 811–833 (2002)
10. Camino, J.F., Helton, J.W., Skelton, R.E., Ye, J.: Matrix inequalities: a symbolic procedure to determine convexity automatically. *Integral Equation and Operator Theory* **46**(4), 399–454 (2003)
11. Cafuta, K., Klep, I., Povh, J.: NCSOStools: a computer algebra system for symbolic and numerical computation with noncommutative polynomials. *Optimization methods and Software* **26**(3), 363–380 (2011)
12. Connes, A.: Classification of injective factors. Cases  $\text{II}_1$ ,  $\text{II}_{\infty}$ ,  $\text{III}_{\lambda}$ ,  $\lambda \neq 1$ . *Annals of Mathematics* **104**, 73–115 (1976)
13. Effros, E.G., Winkler, S.: Matrix convexity: operator analogues of the bipolar and Hahn-Banach theorems. *Journal of Functional Analysis* **144**(1), 117–152 (1997)
14. Hay, D.M., Helton, J.W., Lim, A., McCullough, S.: Non-commutative partial matrix convexity. *Indiana University Mathematics Journal* **57**(6), 2815–2842 (2008)
15. Helton, J.W.: ‘Positive’ noncommutative polynomials are sums of squares. *Annals Of Mathematics* **156**(2), 675–694 (2002)
16. Helton, J.W., de Oliveira, M.C., Stankus, M., Miller, R.L.: NCAlgebra, 2010 release edition. Available from <http://math.ucsd.edu/~ncalg> (2010)
17. Helton, J.W., Klep, I., McCullough, S.: Analytic mappings between noncommutative pencil balls. *Journal of Mathematical Analysis and Applications* **376**(2), 407–428 (2011)
18. Helton, J.W., Klep, I., McCullough, S.: The matricial relaxation of a linear matrix inequality. Preprint, available from <http://arxiv.org/abs/1003.0908>
19. Helton, J.W., Klep, I., McCullough, S.: Proper analytic free maps. *Journal of Functional Analysis* **260**(5), 1476–1490 (2011)
20. Helton, J.W., Klep, I., McCullough, S., Slinglend, N.: Noncommutative ball maps. *Journal of Functional Analysis* **257**(1), 47–87 (2009)
21. Helton, J.W., McCullough, S.: Every free basic convex semi-algebraic set has an LMI representation. Preprint, available from <http://arxiv.org/abs/0908.4352>
22. Helton, J.W., McCullough, S.: Convex noncommutative polynomials have degree two or less. *SIAM Journal on Matrix Analysis and Applications* **25**(4), 1124–1139 (2003)
23. Helton, J.W., McCullough, S.: A positivstellensatz for non-commutative polynomials. *Transactions of The American Mathematical Society* **356**(9), 3721–3737 (2004)

24. Helton, J.W., McCullough, S.A., Putinar, M.: A non-commutative positivstellensatz on isometries. *Journal Für Die Reine Und Angewandte Mathematik* **568**, 71–80 (2004)
25. Helton, J.W., McCullough, S., Putinar, M.: Non-negative hereditary polynomials in a free \*-algebra. *Mathematische Zeitschrift* **250**(3), 515–522 (2005)
26. Helton, J.W., McCullough, S., Putinar, M.: Strong majorization in a free \*-algebra. *Mathematische Zeitschrift* **255**(3), 579–596 (2007)
27. Helton, J.W., McCullough, S.A., Vinnikov, V.: Noncommutative convexity arises from linear matrix inequalities. *Journal Of Functional Analysis* **240**(1), 105–191 (2006)
28. Helton, J.W., Putinar, M.: Positive polynomials in scalar and matrix variables, the spectral theorem, and optimization. In: Bakonyi, M., Gheondea, A., Putinar, M., Rovnyak, J. (eds.) *Operator Theory, Structured Matrices, and Dilations. A Volume Dedicated to the Memory of Tiberiu Constantinescu*, pp. 229–306. Theta, Bucharest (2007)
29. Helton, J.W., Vinnikov, V.: Linear matrix inequality representation of sets. *Communications On Pure And Applied Mathematics* **60**(5), 654–674 (2007)
30. Klep, I., Schweighofer, M.: A nichtnegativstellensatz for polynomials in noncommuting variables. *Israel Journal of Mathematics* **161**(1), 17–27 (2007)
31. Klep, I., Schweighofer, M.: Connes' embedding conjecture and sums of Hermitian squares. *Advances in Mathematics* **217**, 1816–1837 (2008)
32. Klep, I., Schweighofer, M.: Sums of Hermitian squares and the BMV conjecture. *Journal of Statistical Physics* **133**(4), 739–760 (2008)
33. Kaluzhnyi-Verbovetskyi, D.S., Vinnikov, V.: Singularities of rational functions and minimal factorizations: the noncommutative and the commutative setting. *Linear Algebra and its Applications* **430**(4), 869–889 (2009)
34. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization* **11**(3), 796–817 (2001)
35. Lewis, A.S., Parrilo, P.A., Ramana, M.V.: The Lax conjecture is true. *Proceedings of the American Mathematical Society* **133**(9), 2495–2499 (2005)
36. McCullough, S.: Factorization of operator-valued polynomials in several non-commuting variables. *Linear Algebra and its Applications* **326**(1-3), 193–203 (2001)
37. Navascués, M., Pironio, S., Acín, A.: SDP relaxations for non-commutative polynomial optimization. this volume.
38. Parrilo, P.A.: Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization. PhD thesis, California Institute of Technology (2000)
39. Paulsen, V.: Completely bounded maps and operator algebras, Cambridge Studies in Advanced Mathematics, vol. 78. Cambridge University Press, Cambridge (2002)
40. Prestel, A., Delzell, C.N.: Positive polynomials. From Hilbert's 17th problem to real algebra. Springer Monographs in Mathematics. Springer, Berlin (2001)
41. Pisier, G.: Introduction to operator space theory, London Mathematical Society Lecture Note Series, vol. 294. Cambridge University Press, Cambridge (2003)
42. Pironio, S., Navascués, M., Acín, A.: Convergent relaxations of polynomial optimization problems with noncommuting variables. *SIAM Journal on Optimization* **20**(5), 2157–2180 (2010)
43. Popescu, G.: Noncommutative transforms and free pluriharmonic functions. *Advances in Mathematics* **220**(3), 831–893 (2009)
44. Popescu, G.: Free holomorphic functions on the unit ball of  $B(H)^n$ . II. *Journal of Functional Analysis* **258**(5), 1513–1578 (2010)
45. Putinar, M.: Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal* **42**(3), 969–984 (1993)
46. Schmüdgen, K.: A strict Positivstellensatz for the Weyl algebra. *Math. Ann.* **331**(4), 779–794 (2005)
47. Schmüdgen, K., Savchuk, Y.: Unbounded induced representations of \*-algebras. Preprint, <http://arxiv.org/abs/0806.2428>

48. Skelton, R.E., Iwasaki, T., Grigoriadis, K.M.: A unified algebraic approach to linear control design. The Taylor & Francis Systems and Control Book Series. Taylor & Francis Ltd., London (1998)
49. Voiculescu, D.: Free analysis questions II: The Grassmannian completion and the series expansions at the origin. *Journal Für Die Reine Und Angewandte Mathematik* **645**, 155–236 (2010)
50. Voiculescu, D.: Free analysis questions. I. Duality transform for the coalgebra of  $\partial_{X^*B}$ . *International Mathematics Research Notices* **16**, 793–822 (2004)
51. Voiculescu, D.: Aspects of free probability. In: XIVth International Congress on Mathematical Physics, pp. 145–157. World Sci. Publ., Hackensack, NJ (2005)

# Chapter 14

## Positivity and Optimization: Beyond Polynomials

Jean B. Lasserre and Mihai Putinar

### 14.1 Introduction

Polynomial optimization has attracted during the recent period a lot of attention due to the fact that key sums of squares (s.o.s.) representation results from real algebraic geometry combined with semidefinite programming have permitted to define convergent (and efficient) numerical schemes for approximating the global minimum of a polynomial over a compact basic semi-algebraic set. It is not an accident that the same approach also works for solving the *generalized problem of moments* with polynomial data whose important applications seem endless. For more details on this so-called *moment-sos* approach the interested reader is referred to e.g. [32] and the many references therein.

Even though questions involving polynomial data already represent a large framework, many interesting applications involve non-polynomial functions and so it is natural to inquire whether s.o.s.-type positivity certificates can be developed in algebras richer than the polynomial ring so that the moment-s.o.s. approach can be extended to that context. Typical examples are algebras generated by power monomials and some elementary functions composed with them such as the absolute value and fractional powers, trigonometric functions, exponentials, splines or discontinuous step functions. Of special interest both on the algebraic side and for applications to approximation theory is the algebra generated by

---

J.B. Lasserre (✉)

LAAS-CNRS and Institute of Mathematics, University of Toulouse,  
7 Avenue du Colonel Roche, 31077 Toulouse Cédex 4, France  
e-mail: [lasserre@laas.fr](mailto:lasserre@laas.fr)

M. Putinar

Department of Mathematics, University of California at Santa Barbara,  
Santa Barbara, CA, 93106-3080, USA  
e-mail: [mputinar@math.ucsb.edu](mailto:mputinar@math.ucsb.edu)

piecewise polynomial functions, see [5]. Very recently, and in complete harmony with the present article, a positivity certificate for such functions was obtained by Plaumann [41].

**Contribution.** The present article is based on [33]; its aim is to review recent positivity certificates (known also as Positivstellensätze) in the context of algebras generated by some non-polynomial functions, and to provide a robust relaxation method for the numerical verification of the constrained positivity of such functions as well as for optimization purposes. Either we work with semi-algebraic functions, in which case a mere (non-linear) change of coordinates and increase of the number of variables reduces the analysis to a by now classical polynomial optimization problem, or we rely on an approximation of transcendental functions by algebraic ones, and again reduce everything to polynomial optimization. The challenge is to make effective these two different procedures.

We first review general theoretical principles and results of [33] that yield certificates of positivity on a compact set  $\mathbf{K} \subset \mathbb{R}^n$  defined by finitely many inequality constraints involving functions of this algebra. The very recent investigations of Burgdorf et al. [7] and Marshall and Netzer [34] are relevant for positivity questions in algebras of non-polynomial functions. When  $\mathcal{A}$  is the polynomial ring and the set  $\mathbf{K}$  is a compact basic semi-algebraic set, one retrieves a Positivstellensatz due to the second author [42]. We then provide several examples of algebras for which such results apply. In particular:

- The algebra of functions generated by a finite family of functions that contains the polynomials and some elementary functions like e.g. the absolute value, fractional power, and minimum of several polynomials.
- The algebra which consists of functions generated by finitely many basic monadic and dyadic relations on polynomials. The monadic operations are  $|\cdot|$  and  $(\cdot)^{1/p}$ ,  $p = 1, 2, \dots$ , while the dyadic operations are  $(+, \times, /, \wedge, \vee)$ . Notice that this algebra contains highly nonlinear and nondifferentiable functions! We show that this algebra is a subclass of semi-algebraic functions and every element of  $\mathcal{A}$  has the very nice property that it has a *lifted basic semi-algebraic representation*.
- The algebra of functions generated by a finite family that contains the polynomials and some basic transcendental functions like e.g. exponentials and trigonometric polynomials.
- The algebra of piecewise polynomials on a simplicial complex.

We also provide a converging hierarchy of semidefinite relaxations for testing positivity as well as for optimization. Actually, the original optimization problem in the algebra reduces to an equivalent *polynomial* optimization problem in a lifted space. The dimension of the lifting (i.e. the number of additional variables) is directly related to the number of elementary operations on polynomials needed to define the functions in the description of the original problem. Moreover, the resulting polynomial optimization problem exhibits a natural sparsity pattern with respect to the lifted variables for which the important *running intersection property* holds true. Therefore, if on the one hand one has to handle additional variables,

on the other hand this increase in dimension is partly compensated by this sparsity pattern that permits to apply the sparse semidefinite relaxations defined in [50] and whose convergence was proved in [31].

Finally, we also consider the positivity of some non-polynomial functions (e.g. polynomials of exponentials) on *non-compact* sets (e.g. on the real half line). Integral transforms may provide useful information from which partial (and sometimes complete) answers can be derived. In particular, and as another illustrative application of positivity of non-polynomial functions, we consider the stability of trajectories for differential and difference equations with delays, an important issue in control. Again partial (and in some cases complete) answers are provided in terms of positivity certificates.

We hope that, in spite of the incomplete nature of the present tutorial, the questions raised here will have multiple theoretical continuations and an immediate impact on the applied side.

## 14.2 Quadratic Modules and Their Polars

The present section, largely based on [33], is devoted to a brief introduction to the duality framework necessary for the optimization theory applications. Without aiming at completeness, or full generality, we confine ourselves to work with Borel measurable functions on an Euclidean space support. For the real algebra theory invoked below we refer the reader to the monographs [3, 40], while for the most recent and comprehensive algebraic framework devoted to weighted sums of squares decomposition we recommend [7]. The practitioner may skip this section and read directly the examples in the following section.

Let  $X \subset \mathbb{R}^d$  be a Borel measurable set and let  $\mathcal{A}$  be a unital algebra of real valued Borel measurable functions defined on  $X$ . We denote by  $\Sigma\mathcal{A}^2$  the convex cone of squares of elements of  $\mathcal{A}$ . A *quadratic module*  $Q \subset \mathcal{A}$  is a convex cone containing 1, such that

$$(\Sigma\mathcal{A}^2) \cdot Q \subset Q.$$

In practice we deal with *finitely generated quadratic modules*, of the form

$$Q = \Sigma\mathcal{A}^2 + \Sigma\mathcal{A}^2 \cdot h_1 + \dots + \Sigma\mathcal{A}^2 \cdot h_m,$$

where  $h_1, \dots, h_m \in \mathcal{A}$ . The *positivity set* of a quadratic module  $Q$  is by definition

$$P(Q) = \{\mathbf{x} \in X : f(\mathbf{x}) \geq 0, \forall f \in Q\}.$$

Similarly to complex algebraic geometry, the duality between a quadratic module and its positivity set lies at the heart of real algebraic geometry [3, 40]. A useful tool for implementing this duality is provided by the abstract moment problem on the algebra  $\mathcal{A}$ , from where we import a simple definition: we say that the quadratic

module  $Q \subset \mathcal{A}$  possesses the *moment property* if every linear functional  $L \in \mathcal{A}'$  which is non-negative on  $Q$  is represented by a positive Borel measure  $\mu$  supported by  $P(Q)$ .

Also, for duality type arguments, we recall the following concept: an element  $f$  of a convex cone  $C \subset \mathcal{A}$  lies in the *algebraic interior* of  $C$ , if for every  $h \in \mathcal{A}$  there exists  $\epsilon > 0$  with the property  $f + th \in C$  for all  $t, 0 \leq t < \epsilon$ .

The starting point of all representation theorems below is the following simple but crucial observation. Unless otherwise stated, all measurable functions and sets below are meant to be Borel measurable.

**Proposition 14.1.** *Let  $\mathcal{A}$  be a unital algebra of measurable functions defined on the measurable subset  $X \subset \mathbb{R}^d$ . Let  $Q \subset \mathcal{A}$  be a quadratic module with an algebraic interior point and possessing the moment property.*

*If a function  $f \in \mathcal{A}$  is positive on  $P(Q)$ , then  $f \in Q$ .*

*Proof.* Assume by contradiction that  $f \notin Q$ , and let  $\xi$  be an interior point of  $Q$ . By the separation theorem for convex sets, see [25], or for a historical perspective [20, 28], there exists a linear functional  $L \in \mathcal{A}'$  satisfying

$$L(f) \leq 0 \leq L(h), \quad h \in Q, \quad L(\xi) > 0.$$

By assumption there exists a positive Borel measure  $\mu$ , such that

$$L(a) = \int_{P(Q)} ad\mu, \quad a \in \mathcal{A}.$$

Since  $L(\xi) > 0$  we infer that the measure  $\mu$  is non-zero. On the other hand,  $f|_{P(Q)} > 0$  and  $L(f) \leq 0$ , a contradiction.  $\square$

A rather long and diverse series of converse statements, that is conditions imposed on a quadratic module to possess the moment property are known, see for instance the surveys contained in [43].

Thus the main questions we are faced with at this first stage of inquiry are: under which conditions a quadratic module  $Q$  has an interior point and/or possesses the moment property. While the first question has a simple solution, having to do with the boundedness of the positivity set  $P(Q)$ , the second one involves solving an abstract moment problem, and is more delicate, but on the other hand has a long and glorious past, with a wealth of partial results and side remarks [23, 24, 27].

Towards finding solutions to the above two questions we consider only a finitely generated algebra  $\mathcal{A} = \mathbb{R}[h_1, \dots, h_n]$ , where  $h = (h_1, \dots, h_n)$  is an  $n$ -tuple of measurable functions on the set  $X \subset \mathbb{R}^d$ . Let  $\mathbf{y} = (y_1, \dots, y_n)$  be variables, so that, by noetherianity

$$\mathcal{A} \cong \mathbb{R}[\mathbf{y}]/I$$

where  $I$  is a finitely generated ideal of  $\mathbb{R}[\mathbf{y}]$ . The ideal  $I$  which describes all the algebraic relations between generators of the algebra, is *real radical* in the following sense: if  $p_1^2 + \dots + p_k^2 \in I$ , where  $p_1, \dots, p_k \in \mathbb{R}[\mathbf{y}]$ , then  $p_1, \dots, p_k \in I$ , [3]. Indeed, if

$p_1(h)^2 + \dots + p_k(h)^2 = 0$  as functions defined on  $X$ , then  $p_1 \circ h = 0, \dots, p_k \circ h = 0$  in the algebra  $\mathcal{A}$ , for the usual composition “ $\circ$ ” of functions.<sup>1</sup>

**Lemma 14.1.** *Assume that  $\mathcal{A} = \mathbb{R}[h_1, \dots, h_n]$  is a finitely generated algebra of measurable functions and let  $Q \subset \mathcal{A}$  be a quadratic module. If  $1 - (h_1^2 + \dots + h_n^2) \in Q$ , then the constant function 1 belongs to the algebraic interior of  $Q$ .*

For a simple algebraic proof we refer to [40]. Already a first approximative solution to the moment problem associated to  $Q$  is available.

**Proposition 14.2.** *Let  $\mathcal{A} = \mathbb{R}[h_1, \dots, h_n]$  be an algebra of measurable functions defined on the set  $X \subset \mathbb{R}^d$  and let  $L \in \mathcal{A}'$  be a linear functional which is non-negative on the quadratic module  $\Sigma \mathcal{A}^2 + (1 - (h_1^2 + \dots + h_n^2))\Sigma \mathcal{A}^2$ . Then there exists a positive measure  $\nu$ , supported by the unit ball in  $\mathbb{R}^n$ , with the property*

$$L(p(h_1, \dots, h_n)) = \int_{\|\mathbf{y}\| \leq 1} p(\mathbf{y}) d\nu(\mathbf{y}), \quad p \in \mathbb{R}[\mathbf{y}].$$

*Proof.* The familiar Gelfand–Naimark–Segal construction can be invoked at this moment. In short, define an inner product on the polynomial algebra  $\mathbb{R}[\mathbf{y}]$  by

$$\langle p, q \rangle = L(p(h)q(h)), \quad p, q \in \mathbb{R}[\mathbf{y}].$$

Denote by  $J$  the set of null-vectors  $p \in J$  if and only if  $\langle p, p \rangle = 0$ . It is an ideal of  $\mathbb{R}[\mathbf{y}]$  by the Cauchy–Schwarz inequality. The quotient space  $\mathbb{R}[\mathbf{y}]/J$  is endowed then with a non-degenerate inner product structure. Let  $H$  be its Hilbert space completion. The multiplication operators with the variables  $M_i p(\mathbf{y}) = y_i p(\mathbf{y})$  are self-adjoint and bounded, due to the positivity assumption imposed on  $L$ :

$$\langle M_i p, M_i p \rangle = L(h_i^2 p(h)^2) \leq L(p(h)^2) = \langle p, p \rangle.$$

In addition, the operators  $M_i$  mutually commute on the Hilbert space  $H$ . Thus the spectral theorem gives a positive Borel measure  $\nu$ , with the property

$$L(p(h)) = \langle p, 1 \rangle = \langle p(M_1, \dots, M_n)1, 1 \rangle = \int p d\nu, \quad p \in \mathbb{R}[\mathbf{y}].$$

In addition, the support of the measure  $\nu$  is contained in the unit ball, as a consequence of the spherical contraction assumption  $M_1^* M_1 + \dots + M_n^* M_n \leq I$ .  $\square$

Thus, a quadratic module  $Q \subset \mathcal{A} = \mathbb{R}[h_1, \dots, h_n]$  with  $(1 - (h_1^2 + \dots + h_n^2)) \in Q$  satisfies the moment property if the measure  $\nu$  appearing in the above proposition is the push-forward via the map  $h : X \rightarrow \mathbb{R}^n$  of a positive measure  $\mu$  supported by  $P(Q)$ , that is:

$$\int_{\|\mathbf{y}\| \leq 1} p(\mathbf{y}) d\nu(\mathbf{y}) = \int_{P(Q)} p(h(\mathbf{x})) d\mu(\mathbf{x}), \quad p \in \mathbb{R}[\mathbf{y}].$$

---

<sup>1</sup>With  $g : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $f : \mathbb{R} \rightarrow \mathbb{R}$ ,  $\mathbf{x} \mapsto (f \circ g)(\mathbf{x}) := f(g(\mathbf{x}))$ .

A series of sufficient conditions assuring  $\nu = h_*\mu$  in the above sense are discussed next. Note that the measure  $\mu$  above may not be uniquely determined.

Again for  $\mathcal{A} = \mathbb{R}[h_1, \dots, h_n]$  consider a finitely generated quadratic module  $Q = \Sigma \mathcal{A}^2 + g_0(h) \Sigma \mathcal{A}^2 + g_1(h) \Sigma \mathcal{A}^2 + \dots + g_m(h) \Sigma \mathcal{A}^2$ , where  $g_0, \dots, g_m \in \mathbb{R}[\mathbf{y}]$  and  $g_0(\mathbf{y}) = 1 - y_1^2 - \dots - y_n^2$ . Let also the radical ideal of relations for  $\mathcal{A}$  be  $I = (p_1(\mathbf{y}), \dots, p_k(\mathbf{y}))$ . Then

$$\tilde{Q} = \Sigma \mathbb{R}[\mathbf{y}]^2 + g_0(\mathbf{y}) \Sigma \mathbb{R}[\mathbf{y}]^2 + \dots + g_m(\mathbf{y}) \Sigma \mathbb{R}[\mathbf{y}]^2 \pm p_1 \Sigma \mathbb{R}[\mathbf{y}]^2 \pm \dots \pm p_k \Sigma \mathbb{R}[\mathbf{y}]^2$$

is a quadratic module in the polynomial algebra  $\mathbb{R}[\mathbf{y}]$ . Its positivity set  $P(\tilde{Q})$  is contained in the closed unit ball intersection with the zero set  $V(I)$  of the ideal  $I$ . Note that we have a natural pull-back (substitution) map

$$h^* : \mathbb{R}[\mathbf{y}] \longrightarrow \mathcal{A}, \quad h^*(p) = p \circ h, \quad p \in \mathbb{R}[\mathbf{y}],$$

and  $h^*(\tilde{Q}) = Q$ . At the level of positivity sets we obtain

$$h(P(Q)) = P(\tilde{Q}) \cap h(X) \subseteq P(\tilde{Q}),$$

but the inclusion might be strict as the reader can easily verify on simple examples. The following observation is a direct consequence of the equality  $h^*(\tilde{Q}) = Q$ ; we include nevertheless an independent proof.

**Lemma 14.2.** *With the above notation, assume that  $f = p \circ h \in Q$ , where  $p \in \mathbb{R}[\mathbf{y}]$ . Then  $p$  is non-negative on  $P(\tilde{Q})$ .*

*Proof.* The assumption  $f \in Q$  amounts to the algebraic identity

$$f \circ h = \sigma \circ h + (g_0 \circ h)(\sigma_0 \circ h) + \dots + (g_m \circ h)(\sigma_m \circ h),$$

where  $\sigma, \sigma_0, \dots, \sigma_m \in \Sigma \mathbb{R}[\mathbf{y}]^2$ . In its turn, this identity becomes

$$f(\mathbf{y}) = \sigma(\mathbf{y}) + g_0(\mathbf{y})\sigma_0(\mathbf{y}) + \dots + g_m(\mathbf{y})\sigma_m(\mathbf{y}) + r(\mathbf{y}),$$

where  $r \in I$ . By evaluating at a point  $\mathbf{a} \in P(\tilde{Q}) \subset V(I)$  we infer

$$f(\mathbf{a}) = \sigma(\mathbf{a}) + g_0(\mathbf{a})\sigma_0(\mathbf{a}) + \dots + g_m(\mathbf{a})\sigma_m(\mathbf{a}) \geq 0.$$

□

We will see in the next section simple examples showing that the positivity of a function  $f$  on  $P(Q)$  does not guarantee in general a decomposition of the form  $f \in Q$ . The gap between the two statements is explained in the following general observation.

**Theorem 14.1.** *Let  $\mathcal{A} = \mathbb{R}[h_1, \dots, h_n]$  be a finitely generated algebra of measurable functions defined on a measurable set  $X \subset \mathbb{R}^d$  and let  $Q$  be a quadratic module.*

Assume that  $f \in \mathcal{A}$  is positive on  $P(Q)$ . Then  $f \in \hat{Q}$ , where  $\hat{Q}$  is the pullback by the map  $h : X \rightarrow \mathbb{R}^n$  of a quadratic module which serves as a positivity certificate for the set  $h(P(Q))$ .

Of course the choice of  $\hat{Q}$  is not unique, and it may not be a finitely generated quadratic module. The art of finding an appropriate saturation  $\hat{Q}$  is the main ingredient for solving effectively a given non-polynomial optimization problem. For instance, as a first rough approximation, one can choose

$$Q' = \left\{ p \circ h : p|_{h(X) \cap P(\hat{Q})} \geq 0 \right\},$$

where  $\tilde{Q}$  is the pull-back of  $Q$  via the projection map  $\pi : \mathbb{R}[\mathbf{y}] \rightarrow \mathcal{A}$ .

Given a sequence  $\mathbf{z} = (z_\alpha) \subset \mathbb{R}$ ,  $\alpha \in \mathbb{N}^n$ , let  $L_{\mathbf{z}} : \mathbb{R}[\mathbf{y}] \rightarrow \mathbb{R}$  denote the linear mapping

$$f \left( = \sum_{\alpha} f_{\alpha} \mathbf{y}^{\alpha} \right) \mapsto L_{\mathbf{z}}(f) := \sum_{\alpha} f_{\alpha} z_{\alpha}.$$

**Corollary 14.1 ([33]).** Assume, in the conditions of Theorem 14.1, that the set  $h(X)$  is closed. Choose an at most countable set of polynomials  $v_i \in \mathbb{R}[\mathbf{y}]$ ,  $i \in J$ ,  $|J| \leq \infty$ , with the property

$$\{\mathbf{y} \in \mathbb{R}^n : v_i(\mathbf{y}) \geq 0, i \in J\} = h(X) \cap P(\hat{Q}).$$

(a) Every element  $f \circ h \in \mathcal{A}$  which is positive on the set  $P(Q)$  can be written as

$$f \circ h = \sigma \circ h + (M - h_1^2 - \dots - h_n^2)(\sigma_0 \circ h) + \sum_{i \in J_0} (v_i \circ h)(\sigma_i \circ h),$$

where  $M > 0$ , the subset  $J_0 \subset J$  is finite and  $\sigma, \sigma_0, \sigma_i \in \Sigma \mathbb{R}[\mathbf{y}]^2$ ,  $i \in J_0$ .

(b) A sequence  $\mathbf{z} = (z_\alpha)$ ,  $\alpha \in \mathbb{N}^n$ , satisfies

$$z_{\alpha} = \int_{P(Q)} h(\mathbf{x})^{\alpha} d\mu(\mathbf{x}), \quad \forall \alpha \in \mathbb{N}^n$$

if and only if  $\mathbf{z}$  has a representing measure  $\nu$  on  $h(X) \cap P(\hat{Q}) \subset \mathbb{R}^n$ , and equivalently, if and only if,  $L_{\mathbf{z}}(f^2) \geq 0$  and  $L_{\mathbf{z}}(f^2 v_j) \geq 0$  for every  $f \in \mathbb{R}[\mathbf{y}]$  and all  $j \in J$ . In such a case, for every  $\alpha \in \mathbb{N}^n$ ,

$$z_{\alpha} = \int_{h(X) \cap P(\hat{Q})} \mathbf{y}^{\alpha} d\nu(\mathbf{y}) = \int_{P(Q)} h(\mathbf{x})^{\alpha} d\mu(\mathbf{x}).$$

The fact that one always can describe a closed subset of the Euclidean space by countably many polynomial inequalities can be derived from the observation that the complement, that is an open set, is the union of all balls with rational coordinates as

center and rational radii. It is well known that such an accessible (that is, countable number of conditions) description is not available for arbitrary Borel sets.

The remainder of this section is concerned with particular choices of the “saturation” process  $Q \mapsto \hat{Q}$  and the resulting effective sums of squares representations. A detailed analysis, on the theoretical side, of the construction of  $\hat{Q}$  from the character space of a naturally associated real algebra is performed in the recent work [34].

## 14.3 Examples of Positivity Certificates

### 14.3.1 Elementary Semi-algebraic Functions

Let  $X = \mathbb{R}^d$  and let  $\mathcal{A} = \mathbb{R}[\mathbf{x}, h_1(\mathbf{x}), \dots, h_t(\mathbf{x})]$  for some functions  $h_i : \mathbb{R}^d \rightarrow \mathbb{R}$ ,  $i = 1, \dots, t$ . Consider the set  $\mathbf{K} \subset \mathbb{R}^d$  defined by:

$$\mathbf{K} := \{\mathbf{x} \in \mathbb{R}^d : g_j(\mathbf{x}, h_1(\mathbf{x}), \dots, h_t(\mathbf{x})) \geq 0, j = 1, \dots, m\}, \quad (14.1)$$

for some  $(g_j) \subset \mathcal{A}$ , and suppose that  $\mathbf{K}$  is compact and

$$\mathbf{x} \mapsto M - \|\mathbf{x}\|^2 - \sum_{i=1}^t |h_i(\mathbf{x})|^2 > 0 \quad \text{on } \mathbf{K}. \quad (14.2)$$

*Example 14.1.* Here  $h_i(\mathbf{x}) = |x_i|$ ,  $i = 1, \dots, d$  (so let  $|\mathbf{x}| = (|x_i|)$ ). In this case  $\mathbf{y} = (y_1, \dots, y_{2d})$  and one has the algebraic relation  $y_{d+i}^2 = y_i^2$ ,  $i = 1, \dots, d$ , between the generators, hence  $\mathcal{A} \cong \mathbb{R}[\mathbf{y}]/I$ , with  $I := \langle y_{d+1}^2 - y_1^2, \dots, y_{2d}^2 - y_d^2 \rangle$ . Let’s choose the quadratic module with the constant 1 in its algebraic interior:

$$\begin{aligned} Q &= \Sigma \mathcal{A}^2 + \left( M - \left( \sum_{i=1}^d x_i^2 + |x_i|^2 \right) \right) \Sigma \mathcal{A}^2 + \sum_{j=1}^m \Sigma \mathcal{A}^2 g_j(\mathbf{x}, |\mathbf{x}|) \\ &= \Sigma \mathcal{A}^2 + (M - 2\|\mathbf{x}\|^2) \Sigma \mathcal{A}^2 + \sum_{j=1}^m \Sigma \mathcal{A}^2 g_j(\mathbf{x}, |\mathbf{x}|) \end{aligned}$$

The lifted quadratic module is

$$\tilde{Q} = \Sigma \mathbb{R}[\mathbf{y}]^2 + (M - \|\mathbf{y}\|^2) \Sigma \mathbb{R}[\mathbf{y}]^2 + \sum_{j=1}^m \Sigma \mathbb{R}[\mathbf{y}]^2 g_j(\mathbf{y}),$$

whose positivity support set  $P(\tilde{Q})$  is the set

$$\left\{ \mathbf{y} \in \mathbb{R}^{2d} : M - \|\mathbf{y}\|^2 \geq 0; \quad g_j(\mathbf{y}) \geq 0, \quad j = 1, \dots, m \right\}.$$

On the other hand,  $P(Q)$  lives in  $\mathbf{K} \subset \mathbb{R}^d$ . For instance if  $\mathbf{K} = [-1, 1]^d$  and  $t = d$  with  $g_j(\mathbf{x}, |\mathbf{x}|) = 1 - x_j^2$ ,  $j = 1, \dots, d$ , a function such as  $|x_1|^3 + 1/2$  belongs to the algebra  $\mathcal{A}$ , is positive on  $P(Q)$ , but it cannot be written as

$$|x_1|^3 + 1/2 = \sigma(\mathbf{x}, |\mathbf{x}|) + (2d - 2\|\mathbf{x}\|^2)\sigma_0(\mathbf{x}, |\mathbf{x}|) + \sum_{j=1}^d \sigma_j(\mathbf{x}, |\mathbf{x}|)g_j(\mathbf{x}, |\mathbf{x}|),$$

with  $\sigma, \sigma_j \in \Sigma \mathbb{R}[\mathbf{y}]^2$ . Indeed, such a representation would lift to

$$y_{d+1}^3 + 1/2 = \sigma(\mathbf{y}) + (2d - \|\mathbf{y}\|^2)\sigma_0(\mathbf{y}) + \sum_{j=1}^d \sigma_j(\mathbf{y})g_j(\mathbf{y}),$$

and this is obviously impossible by choosing for example  $y_i = 0$  for all  $i \neq d+1$  and  $y_{d+1} = -1$ . In order to obtain the correct representation we invoke the main result of the previous section, and first consider the image set

$$h(X) = \left\{ (\mathbf{x}, |\mathbf{x}|); \mathbf{x} \in \mathbb{R}^d \right\} \subset \mathbb{R}^{2d}.$$

This set can be exactly described by the inequalities

$$y_{d+i}^2 - y_i^2 \geq 0, \quad y_{d+i}^2 - y_i^2 \leq 0, \quad y_{d+i} \geq 0; \quad i = 1, \dots, d.$$

Therefore,

$$\begin{aligned} h(X) \cap P(\tilde{Q}) = & \left\{ \mathbf{y} \in \mathbb{R}^{2d} : 2d - \|\mathbf{y}\|^2 \geq 0; g_j(\mathbf{y}) \geq 0, j = 1, \dots, t; \right. \\ & \left. y_{d+i}^2 - y_i^2 = 0, y_{d+i} \geq 0, i = 1, \dots, d \right\}, \end{aligned}$$

and by Corollary 14.1, the quadratic module in  $\mathbb{R}[\mathbf{y}]$  generated by the polynomials that describe the above semi-algebraic set  $h(X) \cap P(\tilde{Q})$ , provides the desired certificate of positivity. And so we have:

**Proposition 14.3.** *Let  $\mathbf{K} \subset \mathbb{R}^d$  be as in (14.1) and such that (14.2) holds. An element  $f$  of the algebra  $\mathbb{R}[\mathbf{x}, |\mathbf{x}|]$  which is positive on  $\mathbf{K}$  can be written as*

$$\begin{aligned} \mathbf{x} \mapsto f(\mathbf{x}, |\mathbf{x}|) = & \sigma(\mathbf{x}, |\mathbf{x}|) + (M - 2\|\mathbf{x}\|^2)\sigma_0(\mathbf{x}, |\mathbf{x}|) \\ & + \sum_{i=1}^d |x_i| \sigma_i(\mathbf{x}, |\mathbf{x}|) + \sum_{j=1}^t g_j(\mathbf{x}, |\mathbf{x}|) \psi_j(\mathbf{x}, |\mathbf{x}|) \end{aligned}$$

where  $\sigma, \sigma_i, \psi_j \in \Sigma \mathbb{R}[\mathbf{y}]^2$ .

*Example 14.2.* Given two polynomials  $p, q \in \mathbb{R}[\mathbf{x}]$ , let  $\mathcal{A} := \mathbb{R}[\mathbf{x}, p(\mathbf{x}) \vee q(\mathbf{x})]$  where  $a \vee b := \max[a, b]$ , and let  $\mathbf{K}$  be as in (14.1) with  $t = 1$  and  $h_1 = p(\mathbf{x}) \vee q(\mathbf{x})$ . Recall that  $2(a \vee b) = |a - b| + (a + b)$ , and so one may consider the new algebra  $\mathcal{A}' = \mathbb{R}[\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|]$  since a polynomial in the variables  $\mathbf{x}$  and  $|p(\mathbf{x}) - q(\mathbf{x})|$  is a particular polynomial in  $\mathbf{x}$  and  $|p(\mathbf{x}) - q(\mathbf{x})|$ . In this case  $\mathbf{y} = (y_1, \dots, y_{d+1})$  and one has the algebraic dependency

$$y_{d+1}^2 = (p(\mathbf{x}) - q(\mathbf{x}))^2$$

and the additional constraint  $y_{d+1} \geq 0$ .

**Proposition 14.4.** *Let  $\mathbf{K} \subset \mathbb{R}^d$  be as in (14.1) and such that (14.2) holds. Let  $f$  be an element of the algebra  $\mathcal{A} := \mathbb{R}[\mathbf{x}, p(\mathbf{x}) \vee q(\mathbf{x})]$  which is positive on  $\mathbf{K}$ . Equivalently*

$$\mathbf{x} \mapsto f(\mathbf{x}, p(\mathbf{x}) \vee q(\mathbf{x})) = \tilde{f}(\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|),$$

for some  $\tilde{f} \in \tilde{\mathcal{A}} := \mathbb{R}[\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|]$ , positive on  $\mathbf{K}$ . Then  $\tilde{f}$  can be written

$$\begin{aligned} \tilde{f}(\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|) &= \sigma(\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|) \\ &\quad + (M - \|\mathbf{x}\|^2 - |p(\mathbf{x}) - q(\mathbf{x})|^2) \sigma_0(\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|) \\ &\quad + |p(\mathbf{x}) - q(\mathbf{x})| \sigma_1(\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|) \\ &\quad + \sum_{j=1}^m \psi_j(\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|) g_j(\mathbf{x}, |p(\mathbf{x}) - q(\mathbf{x})|) \end{aligned}$$

where  $\sigma, \sigma_i, \psi_j \in \Sigma \mathbb{R}[y_1, \dots, y_{d+1}]^2$ .

Of course a similar thing can be done with  $p(\mathbf{x}) \wedge q(\mathbf{x}) := \min[p(\mathbf{x}), q(\mathbf{x})]$  using  $2(a \wedge b) = (a + b) - |a - b|$ .

*Example 14.3.* Let  $\mathbf{u} = (u_i)_{i=1}^d \in \mathbb{R}[\mathbf{x}]^d$  and let  $f$  be an element of the algebra  $\mathcal{A} := \mathbb{R}[\mathbf{x}, \|\mathbf{u}(\mathbf{x})\|_p]$  which is positive on  $\mathbf{K}$ , where  $\|\mathbf{x}\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$ , with either  $p \in \mathbb{N}$  or  $p^{-1} \in \mathbb{N}$ . For instance, if  $p = 2q + 1$  with  $q \in \mathbb{N}$ , use the algebraic lifting  $\mathbf{y} \in \mathbb{R}^{2d+1}$ , with:

$$y_{d+i}^2 = u_i(\mathbf{x})^2; \quad y_{2d+1}^p = \sum_{i=1}^n y_{d+i}^p; \quad y_{d+i} \geq 0, i = 1, \dots, d+1,$$

to obtain:

**Proposition 14.5.** *Let  $\mathbf{K}$  be as in (14.1) and such that (14.2) holds. Let  $f$  be an element of the algebra  $\mathcal{A} := \mathbb{R}[\mathbf{x}, \|\mathbf{u}(\mathbf{x})\|_p]$  which is positive on  $\mathbf{K}$ . If  $p = 2q + 1$  with  $q \in \mathbb{N}$  then  $f$  can be written as:*

$$\begin{aligned} f(\mathbf{x}, \|\mathbf{u}(\mathbf{x})\|_p) &= \sigma(\mathbf{x}, |\mathbf{u}(\mathbf{x})|, \|\mathbf{u}(\mathbf{x})\|_p) + \sum_{j=1}^m g_j(\mathbf{x}) \psi_j(\mathbf{x}, |\mathbf{u}(\mathbf{x})|, \|\mathbf{u}(\mathbf{x})\|_p) \\ &\quad + (M - \|\mathbf{x}\|^2 - \|\mathbf{u}(\mathbf{x})\|^2 - \|\mathbf{u}(\mathbf{x})\|_p^2) \sigma_0(\mathbf{x}, |\mathbf{u}(\mathbf{x})|, \|\mathbf{u}(\mathbf{x})\|_p) \end{aligned}$$

$$+ \sum_{i=1}^d |u_i(\mathbf{x})| \sigma_i(\mathbf{x}, |\mathbf{u}(\mathbf{x})|, \|\mathbf{u}(\mathbf{x})\|_p) \\ + \|\mathbf{u}(\mathbf{x})\|_p \phi(\mathbf{x}, |\mathbf{u}(\mathbf{x})|, \|\mathbf{u}(\mathbf{x})\|_p)$$

where  $\sigma, \sigma_i, \psi_j, \phi \in \Sigma \mathbb{R}[y_1, \dots, y_{2d+1}]^2$ .

To obtain a similar and appropriate representation if  $p/2 \in \mathbb{N}$ , use the algebraic lifting  $\mathbf{y} \in \mathbb{R}^{d+1}$  with:

$$y_{d+1}^p = \sum_{i=1}^d u_i(\mathbf{x})^p; \quad y_{d+1} \geq 0.$$

And if  $1/p \in \mathbb{N}$  then use the algebraic lifting  $\mathbf{y} \in \mathbb{R}^{d+1}$  with:

$$y_{d+i}^{2/p} = u_i(\mathbf{x})^2; \quad y_{2d+1} = \left( \sum_{i=1}^n y_{d+i} \right)^{1/p}; \quad y_{d+i} \geq 0, i = 1, \dots, d+1.$$

*Example 14.4.* Let  $\mathcal{A} = \mathbb{R}[\mathbf{x}, \sqrt{p(\mathbf{x})}]$  for some polynomial  $p \in \mathbb{R}[\mathbf{x}]$ . Of course one here considers the new basic semi-algebraic set  $\mathbf{K}' \subset \mathbb{R}^d$  defined by:

$$\mathbf{K}' := \mathbf{K} \cap \{\mathbf{x} : g_{m+1}(\mathbf{x}) \geq 0\} \quad \text{with} \quad \mathbf{x} \mapsto g_{m+1}(\mathbf{x}) := p(\mathbf{x}), \quad (14.3)$$

and the lifting  $y_{d+1}^2 = p(\mathbf{x})$ , with the constraint  $y_{d+1} \geq 0$ .

**Proposition 14.6.** *Let  $f$  be an element of the algebra  $\mathcal{A} := \mathbb{R}[\mathbf{x}, \sqrt{p(\mathbf{x})}]$  which is positive on  $\mathbf{K}'$ . Then  $f$  can be written*

$$f(\mathbf{x}, \sqrt{p(\mathbf{x})}) = \sigma(\mathbf{x}, \sqrt{p(\mathbf{x})}) + (M - \|\mathbf{x}\|^2 - p(\mathbf{x})) \sigma_0(\mathbf{x}, \sqrt{p(\mathbf{x})}) \\ + \sum_{j=1}^{m+1} g_j(\mathbf{x}) \psi_j(\mathbf{x}, \sqrt{p(\mathbf{x})}) + \sqrt{p(\mathbf{x})} \phi(\mathbf{x}, \sqrt{p(\mathbf{x})})$$

where  $\sigma, \sigma_i, \phi, \psi_j \in \Sigma \mathbb{R}[y_1, \dots, y_{d+1}]$ .

The natural adjunction of square roots of polynomials in Positivstellensätze goes back at least to [36].

### 14.3.2 A More General Class of Semi-algebraic Functions

We now consider an algebra of functions which forms an important subclass of semi-algebraic functions. Recall the notation

$$a \vee b = \max[a, b]; \quad a \wedge b := \min[a, b]; \quad |\mathbf{x}| = (|x_1|, \dots, |x_d|) \in \mathbb{R}^d,$$

and let  $\mathcal{A}$  be the algebra of functions  $f : \mathbf{K} \rightarrow \mathbb{R}$  generated by finitely many of the dyadic operations  $\{+, \times, /, \vee, \wedge\}$  and monadic operations  $|\cdot|$  and  $(\cdot)^{1/p}$  ( $p \in \mathbb{N}$ ) on polynomials. For instance, with  $a, \ell \in \mathbb{R}[\mathbf{x}]$  and  $p = (p_1, p_2) \in \mathbb{R}[\mathbf{x}]^2$ , the function

$$\mathbf{x} \mapsto f(\mathbf{x}) := a(\mathbf{x}) \wedge \left( \sqrt{\|p(\mathbf{x})\|_{2q+1} + \ell(\mathbf{x})} \right) \quad (14.4)$$

is a member of  $\mathcal{A}$  (e.g. assuming that  $\ell(\mathbf{x}) \geq 0$  for all  $\mathbf{x} \in \mathbf{K}$ ). Let  $(g_j)_{j=1}^m \subset \mathbb{R}[\mathbf{x}]$ , let  $\mathbf{K} \subset \mathbb{R}^d$  be the basic semi-algebraic set

$$\mathbf{K} := \{\mathbf{x} \in \mathbb{R}^d : g_j(\mathbf{x}) \geq 0, j = 1, \dots, m\} \quad (14.5)$$

and suppose that  $\mathbf{K}$  is compact and

$$\mathbf{x} \mapsto M - \|\mathbf{x}\|^2 > 0 \quad \text{on } \mathbf{K}. \quad (14.6)$$

Recall that  $f : \mathbf{K} \rightarrow \mathbb{R}$  is a semi-algebraic function if its graph  $\Psi_f := \{(\mathbf{x}, f(\mathbf{x})) : \mathbf{x} \in \mathbf{K}\}$  is a semi-algebraic set of  $\mathbb{R}^d \times \mathbb{R}$ . Motivated by the efficiency of the constructive aspects of non-polynomial optimization, we adopt the following ad-hoc terminology.

**Definition 14.1.** A function  $f \in \mathcal{A}$  is said to have a *basic semi-algebraic lifting* (in short a “b.s.a.l.”), or  $f$  is *basic semi-algebraic* (b.s.a.) if there exist  $p, s \in \mathbb{N}$ , polynomials  $(h_k)_{k=1}^s \subset \mathbb{R}[\mathbf{x}, y_1, \dots, y_p]$  and a basic semi-algebraic set

$$\mathbf{K}_f := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{d+p} : \mathbf{x} \in \mathbf{K}; h_k(\mathbf{x}, \mathbf{y}) \geq 0, k = 1, \dots, s\} \quad (14.7)$$

such that the graph of  $f$  (denoted  $\Psi_f$ ) satisfies:

$$\Psi_f := \{(\mathbf{x}, f(\mathbf{x})) : \mathbf{x} \in \mathbf{K}\} = \{(\mathbf{x}, y_p) : (\mathbf{x}, \mathbf{y}) \in \mathbf{K}_f\}. \quad (14.8)$$

In other words,  $\Psi_f$  is an orthogonal projection of the basic semi-algebraic set  $\mathbf{K}_f$  which lives in the lifted space  $\mathbb{R}^{d+p}$ .

Hence by the projection theorem of real algebraic geometry [3], a function  $f \in \mathcal{A}$  that admits a b.s.a.l. is semi-algebraic. As a matter of fact, *every* semi-algebraic function possesses a basic semi-algebraic lifting.<sup>2</sup> This can be deduced from the fact that every semi-algebraic set  $S$  is the projection of a closed, basic semi-algebraic set. Indeed, assume first that  $S$  is closed and  $S = \cup_{i \in I} S_i$  where every  $S_i$  is closed, basic semi-algebraic and the set  $I$  is finite. Writing

$$S_i = \{\mathbf{x} \in \mathbb{R}^n ; g_1(\mathbf{x}) \geq 0, \dots, g_k(\mathbf{x}) \geq 0\}$$

---

<sup>2</sup>The authors wish to thank D. Plaumann for pointing out this fact.

we can lift the inequalities to equalities as before:

$$S_i = \{\mathbf{x} \in \mathbb{R}^n; \exists \mathbf{y} \in \mathbb{R}^k, y_i^2 - g_i(\mathbf{x}) = 0, i = 1, \dots, k\},$$

and remark that a union of real algebraic sets is real algebraic. The same procedure works for an open component of an arbitrary semi-algebraic set  $S$ :

$$G_j = \{\mathbf{x} \in \mathbb{R}^n; h_1(\mathbf{x}) > 0, \dots, h_k(\mathbf{x}) > 0\},$$

can be represented as

$$G_j = \{\mathbf{x} \in \mathbb{R}^n; \exists \mathbf{y} \in \mathbb{R}^k, 1 - y_i^2 h_i(\mathbf{x}) = 0 \text{ } i = 1, \dots, k\}.$$

**Lemma 14.3 ([33]).** *Let  $\mathbf{K}$  be the basic semi-algebraic set in (14.5). Then every well-defined  $f \in \mathcal{A}$  has a basic semi-algebraic lifting.*

*Example 14.5.* With  $f \in \mathcal{A}$  as in (14.4),  $\Psi_f = \{(\mathbf{x}, y_6) : (\mathbf{x}, \mathbf{y}) \in \mathbf{K}_f\}$ , where  $\mathbf{K}_f \subset \mathbb{R}^{d+6}$  is the basic semi-algebraic set:

$$\begin{aligned} \mathbf{K}_f = \{&(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{d+6} : g_j(\mathbf{x}) \geq 0, j = 1, \dots, m \\ &y_i^2 - p_i(\mathbf{x})^2 = 0, i = 1, 2; y_3^{2q+1} - y_1^{2q+1} - y_2^{2q+1} = 0; \\ &y_4^2 - \ell(\mathbf{x}) - y_3 = 0; y_5^2 - (a(\mathbf{x}) - y_4)^2 = 0; \\ &2y_6 - (a(\mathbf{x}) + y_4) + y_5 = 0; y_1, y_2, y_4, y_5 \geq 0\}. \end{aligned}$$

Observe that with each variable  $y_k$  of the lifting  $\mathbf{y} = (y_1, \dots, y_p)$  is associated a certain function  $v_k \in \mathcal{A}$ . For instance in Example 14.5:

$$y_i \rightarrow v_i(\mathbf{x}) := |p_i(\mathbf{x})|, i = 1, 2; y_3 \rightarrow v_3(\mathbf{x}) := (v_1(\mathbf{x})^{2q+1} + v_2(\mathbf{x})^{2q+1})^{1/(2q+1)}$$

$$y_4 \rightarrow v_4(\mathbf{x}) := \sqrt{v_3(\mathbf{x}) + \ell(\mathbf{x})}; y_5 \rightarrow v_5(\mathbf{x}) := |a(\mathbf{x}) - v_4(\mathbf{x})|$$

and  $y_6 \rightarrow v_6(\mathbf{x}) := f(\mathbf{x})$ .

*Remark 14.1.* In fact, we have just seen that for the algebra  $\mathcal{A}$  that we consider, the constraints  $h_k(\mathbf{x}, \mathbf{y}) \geq 0, k = 1, \dots, s$ , in the definition (14.7) of  $\mathbf{K}_f$ , decompose into:

- Equality constraints  $u_\ell(\mathbf{x}, \mathbf{y}) = 0$  that describe algebraic relationships,
- Inequality constraints of the form  $y_j \geq 0, j \in J$  (for some index set  $J$ ), needed for a complete description of the b.s.a. function  $f$ .

Notice that if  $\mathbf{v} = (v_k) \subset \mathcal{A}$  denotes the intermediate b.s.a. functions associated with the lifting  $\mathbf{y}$ , then  $u_\ell(\mathbf{x}, \mathbf{v}(\mathbf{x})) = 0$  for every  $\ell$ .

Next, with  $\mathbf{K} \subset \mathbb{R}^n$  as in (14.5), consider the set  $\mathbf{S} \subseteq \mathbf{K}$  defined by:

$$\mathbf{S} := \{\mathbf{x} \in \mathbf{K} : h_\ell(\mathbf{x}) \geq 0, \ell = 1, \dots, s\} \quad (14.9)$$

for some finite family  $(h_k)_{k=1}^s \subset \mathcal{A}$ .

**Lemma 14.4 ([33]).** *The set  $\mathbf{S}$  in (14.9) is a semi-algebraic set which is the projection of a lifted basic semi-algebraic set. It can be written*

$$\mathbf{S} = \left\{ \mathbf{x} \in \mathbb{R}^d : \exists \mathbf{y} \in \mathbb{R}^t \text{ s.t. } \mathbf{x} \in \mathbf{K}; \quad u_k(\mathbf{x}, \mathbf{y}) = 0, \quad k = 1, \dots, r \right. \\ \left. y_j \geq 0, \quad j \in J \right\} \quad (14.10)$$

for some integers  $r, t$ , some polynomials  $(u_k)_{k=1}^r \subset \mathbb{R}[\mathbf{x}, \mathbf{y}]$  and some index set  $J \subseteq \{1, \dots, t\}$ . With the lifting  $\mathbf{y} = (y_1, \dots, y_t)$  is associated a vector of functions  $\mathbf{v} = (v_1, \dots, v_t) \in \mathcal{A}^t$ .

So in (14.10) the equations  $u_k(\mathbf{x}, \mathbf{y}) = 0$  are the algebraic relationships that link the b.s.a. functions  $v_k \in \mathcal{A}$  with each other and with  $\mathbf{x}$ . See for instance the six equations that define the set  $\mathbf{K}_f$  in Example 14.5. In particular,  $u_k(\mathbf{x}, \mathbf{v}(\mathbf{x})) = 0$  for all  $\mathbf{x} \in \mathbf{K}$ . The additional inequalities  $y_j \geq 0$ ,  $j \in J$ , completely characterize the functions  $v_k \in \mathcal{A}$ . See Remark 14.1.

**Theorem 14.2 ([33]).** *Let  $\mathbf{K} \subset \mathbb{R}^d$  in (14.5) be compact and let  $\mathbf{S}$  be as in (14.9) for some finite family  $(h_\ell) \subset \mathcal{A}$ . Let  $f \in \mathcal{A}$  have the b.s.a.l.*

$$\Psi_f = \left\{ (\mathbf{x}, z_{m_f}) : \mathbf{x} \in \mathbf{K}; q_\ell(\mathbf{x}, \mathbf{z}) = 0, \ell = 1, \dots, t_f; z_j \geq 0, \quad j \in J_f \right\}$$

(where  $\mathbf{z} = (z_1, \dots, z_{m_f})$  and  $J \subseteq \{1, \dots, m_f\}$ ), and let  $\mathbf{v} = (v_1, \dots, v_t) \in \mathcal{A}^t$  (resp.  $\mathbf{p} = (p_1, \dots, p_{m_f}) \in \mathcal{A}^{m_f}$ ) be the vector of b.s.a. functions associated with the lifting  $\mathbf{y}$  in (14.10) (resp. the lifting  $\mathbf{z}$ ). If  $f$  is positive on  $\mathbf{S}$  then

$$f(\mathbf{x}) = \sigma_0(\mathbf{x}, \mathbf{v}(\mathbf{x}), \mathbf{p}(\mathbf{x})) + \sum_{j=1}^m \sigma_j(\mathbf{x}, \mathbf{v}(\mathbf{x}), \mathbf{p}(\mathbf{x})) g_j(\mathbf{x}) \\ + \sum_{k \in J} \psi_k(\mathbf{x}, \mathbf{v}(\mathbf{x}), \mathbf{p}(\mathbf{x})) v_k(\mathbf{x}) + \sum_{\ell \in J_f} \phi_\ell(\mathbf{x}, \mathbf{v}(\mathbf{x}), \mathbf{p}(\mathbf{x})) p_\ell(\mathbf{x}) \\ + \phi_0(\mathbf{x}, \mathbf{v}(\mathbf{x}), \mathbf{p}(\mathbf{x}))(M - \|(\mathbf{x}, \mathbf{v}(\mathbf{x}), \mathbf{p}(\mathbf{x}))\|^2) \quad (14.11)$$

for some  $(\sigma_j), (\psi_k), (\phi_\ell) \subset \Sigma \mathbb{R}[\mathbf{x}, \mathbf{y}, \mathbf{z}]^2$  and a sufficiently large  $M > 0$ .

Of course, all positivity certificates of Sect. 14.3.1 which are specific to a certain type of b.s.a. functions, are a particular case of the more general Theorem 14.2. In the latter, and in contrast with Propositions 14.3–14.6, the lifting (and so the dimension of the vectors  $\mathbf{y}$  and  $\mathbf{z}$ ) depends on the particular functions  $f, h_\ell \in \mathcal{A}$  involved.

### 14.3.3 Some Non Semi-algebraic Functions

*Example 14.6.* Let  $X = [0, \pi/2]$  and  $\mathcal{A} = \mathbb{R}[x, \sin x]$ . Again  $\mathbf{y} = (y_1, y_2)$  but this time the algebra  $\mathcal{A}$  is isomorphic to the full polynomial algebra  $\mathbb{R}[\mathbf{y}]$ . For illustration we consider the quadratic module  $Q \subset \mathcal{A}$  generated by the elements  $x, \pi/2 - x, 1 - \sin^2 x$ . Although the inequality  $1 - \sin^2 x \geq 0$  is redundant, it is necessary to add it for having the function 1 in the algebraic interior of  $Q$ .

As we wish to obtain a certificate of positivity for a function  $f(x, \sin x)$  belonging to  $\mathcal{A}$ , an algebraic description of the graph of  $x \mapsto \sin x$  is in order. Choose the polynomials

$$x \mapsto p_k(x) = \sum_{j=0}^k (-1)^j \frac{x^{2j+1}}{(2j+1)!}, \quad k \geq 0.$$

It is well known that

$$p_{2k}(x) \geq p_{2k+2}(x) \geq \sin x \geq p_{2k+1}(x) \geq p_{2k-1}(x), \quad \forall x \in [0, \pi/2],$$

and that these polynomials converge uniformly to  $\sin x$  on  $[0, \pi/2]$ . A description of the graph of  $\sin x$  is:

$$\begin{aligned} h(X) &= \{(x, \sin x) : 0 \leq x \leq \pi/2\} \\ &= \{\mathbf{y} \in \mathbb{R}^2 : 0 \leq y_1 \leq \pi/2, \quad p_{2k+1}(y_1) \leq y_2 \leq p_{2k}(y_1), \quad k \geq 0\}. \end{aligned}$$

In view of our main result, the conclusion is:

**Proposition 14.7.** *Let  $f$  be an element of the algebra  $\mathbb{R}[x, \sin x]$  and assume that  $f > 0$  for all  $x \in [0, \pi/2]$ . Then there exists  $k \geq 0$  such that*

$$\begin{aligned} x \mapsto f(x, \sin x) &= \sigma(x, \sin x) + x\sigma_0(x, \sin x) + (\pi/2 - x)\sigma_2(x, \sin x) \\ &\quad + [p_{2k}(x) - \sin x]\sigma_3(x, \sin x) + [\sin x - p_{2k+1}(x)]\sigma_4(x, \sin x), \end{aligned}$$

where  $\sigma, \sigma_i \in \Sigma \mathbb{R}[\mathbf{y}]^2$ .

*Example 14.7.* Let  $X = \mathbb{R}$  and  $\mathcal{A} = \mathbb{R}[x, e^{ax}, e^{bx}]$ . Two distinct cases, corresponding to  $a, b$  commensurable or not, will be treated. This dichotomy is well known to the experts in the control theory of delay systems.

To fix ideas, we assume that  $b > a > 0$  and the base is the interval  $[0, 1]$ . Denote  $y_1 = x, y_2 = e^{ax}, y_3 = e^{bx}$ . We will impose the archimedeanity constraints

$$x \geq 0, 1 - x \geq 0, \quad y_2 \geq 0, e^a - y_2 \geq 0, \quad y_3 \geq 0, e^b - y_3 \geq 0.$$

Along with these we have to consider the polynomial approximations of the two exponentials:

$$p_n(ax) = \sum_{k=0}^n \frac{a^k x^k}{k!} \leq y_2 \leq p_n(ax) + \frac{a^{n+1} e^a x^{n+1}}{(n+1)!}$$

and

$$p_n(bx) = \sum_{k=0}^n \frac{b^k x^k}{k!} \leq y_3 \leq p_n(bx) + \frac{b^{n+1} e^b x^{n+1}}{(n+1)!}.$$

The reader will recognize above an upper bound of the remainder of Taylor's series approximation of the exponential function.

*Case I. The frequencies  $a, b$  are not commensurable.* Then there is no algebraic relation among the variables  $y_1, y_2, y_3$  and a function  $f \in \mathbb{R}[x, e^{ax}, e^{bx}]$  which is positive on the interval  $[0, 1]$  is decomposable into a weighted sum of squares generated by  $x, 1 - x, e^{ax}, e^a - e^{ax}, e^{bx}, e^b - e^{bx}, e^{ax} - p_n(ax), p_n(ax) + \frac{a^{n+1} e^a x^{n+1}}{(n+1)!} - e^{ax}, e^{bx} - p_n(bx), p_n(bx) + \frac{b^{n+1} e^b x^{n+1}}{(n+1)!} - e^{bx}$ , for a large enough  $n$ .

Note that the above list of weights is redundant, as for instance the constraint  $e^{ax} > 0$  follows from  $e^{ax} - p_n(ax) > 0$ , and so on. In practice it will be important to identify a minimal set of positivity constraints.

*Case II. The frequencies  $a, b$  are commensurable.* Assume that  $Na = Mb$  where  $N, M$  are positive integers. Then the above decomposition of  $f \in \mathbb{R}[x, e^{ax}, e^{bx}]$  positive on  $[0, 1]$ , holds modulo the algebraic relation  $y_2^N = y_3^M$  and the whole question reduces to a polynomial algebra setting.

*Example 14.8.* When dealing with trigonometric (real valued) polynomials in  $n$  variables, one has to consider the algebra  $\mathcal{A} = \mathbb{R}[\sin \theta_i, \cos \theta_i]$ ,  $1 \leq i \leq d$ . The standard lifting proposed in the present article is

$$y_i = \sin \theta_i, \quad z_i = \cos \theta_i, \quad 1 \leq i \leq d,$$

modulo the ideal generated by the relations

$$y_i^2 + z_i^2 - 1 = 0, \quad 1 \leq i \leq d. \tag{14.12}$$

Note that in this case every quadratic module  $Q \subset \mathcal{A}$  in the variables  $\sin \theta_i, \cos \theta_i$  lifts to an archimedean quadratic module  $\hat{Q}$  in the variables  $y_i, z_i$ . Specifically, relations (14.12) assure that the constant function 1 belongs to the algebraic interior of  $\hat{Q}$ .

#### 14.3.4 Examples Derived from Integral Transforms

In some applications (e.g. in control) one frequently encounters functions that are linear combinations of exponentials and one would like to certify their positivity on some domain (e.g. a non compact interval of the real line). To handle such

a situation, a great deal of information is available when taking integral transforms of (positive) measures, such as the Fourier-Laplace transform or the Hankel transform. We confine ourselves to state below a couple of results which are relevant for the present work.

**Theorem 14.3.** *Let  $0 < a_1 < a_2 < \dots < a_n$  and  $c_1, \dots, c_n$  be real numbers. The exponential polynomial*

$$\sum_{j=1}^n c_j \exp(-a_j x), \quad x \geq 0,$$

*is non-negative if and only if  $\sum_{j=1}^n \frac{c_j}{(s+a_j)^k} \geq 0$  for all  $k \geq 0$  and  $s \geq 0$ .*

*Proof.* Let us consider the positive Borel measure  $d\mu = \sum c_j \exp(-a_j x) dx$ . Its Laplace transform

$$\mathcal{L}\mu(s) = \int_0^\infty e^{-sx} d\mu(x)$$

is well defined for all  $s \geq 0$ , and according to Bernstein's Theorem (see [1])

$$(-1)^k \frac{d^k}{ds^k} \mathcal{L}\mu(s) \geq 0, \quad k \geq 0, \quad s \geq 0.$$

And vice-versa. The complete monotonicity condition becomes:

$$\sum_{j=1}^n \frac{c_j}{(s+a_j)^k} \geq 0, \quad k \geq 0, \quad s \geq 0.$$

□

**Corollary 14.2.** *Assume that  $0 < a_1 < a_2 < \dots < a_n$ ,  $c_1, \dots, c_n$  are real numbers and that  $\sum_{j=1}^n c_j t^{a_j} \geq 0$  for all  $t \in [0, 1]$ . Then, and only then, for every integer  $k \geq 0$ , the rational fraction  $s \mapsto g_k(s) := \sum_{j=1}^n \frac{c_j}{(s+a_j)^k}$  is nonnegative on  $[0, \infty)$ .*

Note that the coefficients  $c_j$  above may not all be non-negative. For instance, the function  $2e^{-x} - e^{-2x} - e^{-\pi x}$  is non-negative on the positive semi-axis. The advantage of the above computation lies in the reduction of the positivity of an exponential polynomial to that of a sequence of rational functions. Theoretically, for fixed  $k \in \mathbb{N}$ , checking whether  $g_k \geq 0$  on  $[0, \infty)$  can be done by solving a single semidefinite program. Namely, reduce to same denominator to obtain a single rational fraction  $p/q$ , with  $p, q \in \mathbb{R}[s]$  and maximize  $\lambda$  such that  $p - \lambda q = \sigma_0 + \sigma_1 s$  for some polynomials  $\sigma_0, \sigma_1 \in \Sigma \mathbb{R}[s]^2$  whose degrees are bounded by that of  $p$  and  $q$ ; see e.g. [32]. Hence if one suspects that the condition of Corollary 14.2 fails, one may try to solve such semidefinite programs with increasing values of  $k$  until  $\lambda_k^* < 0$  for some  $k$ .

Proceeding differently, sufficient conditions of positivity for the function  $\sum_{j=1}^n c_j \exp(-a_j x)$  can be obtained as indicated in the introduction: namely, denote  $y_j = \exp(-a_j x)$  and remark that the inequalities  $1 \geq y_1 \geq y_2 \geq \dots \geq y_n \geq 0$  hold true.

Hence the condition

$$\min_{\mathbf{y}} \left\{ \sum_{j=1}^n c_j y_j : 1 \geq y_1 \geq y_2 \geq \dots \geq y_n \geq 0 \right\} \geq 0$$

provides sufficient conditions on the coefficients  $c_j$  and can be tested by solving a linear program.

On the other hand, if the multipliers  $a_1, \dots, a_n$  are commensurable over  $\mathbb{Q}$ , then a simplification of the above positivity certificate is available. Indeed, assume that  $a_j = \alpha_j a$ ,  $j = 1, \dots, n$ , where  $(\alpha_j)$  are positive integers and  $a > 0$ . Then  $\sum_{j=1}^n c_j \exp(-a_j x)$  for all  $x \geq 0$  if and only if  $\sum_{j=1}^n c_j s^{\alpha_j} \geq 0$  for all  $s \in [0, 1]$ . That is, if and only if  $\sum_{j=1}^n c_j s^{\alpha_j} = \sigma_0 + \sigma_1 s(1-s)$ , for some sums of squares polynomials  $\sigma_0, \sigma_1 \in \Sigma\mathbb{R}[s]^2$  of degree at most  $\max_j \alpha_j$ ; see e.g. [32].

One can ask a similar question for almost periodic functions of the form

$$f(x) = \sum_{k=1}^n c_k \exp(i\lambda_k x), \quad x \in \mathbb{R}.$$

Recall that a function  $\xi \mapsto \phi(\xi)$  is *positive definite* if the matrix  $(\phi(\xi_i - \xi_j))_{i,j}$  is positive semi-definite for all finite systems  $(\xi_i)$  of points on the real line. In view of Bochner's characterization of Fourier transforms of positive measures on the line, we can state the following criterion.

**Proposition 14.8.** *The inequality  $\sum_{k=1}^n c_k \exp[i\lambda_k x] \geq 0$  holds for all  $x \in \mathbb{R}$  if and only if the function*

$$\xi \mapsto F(\xi) := \sum_{k=1}^n c_k \exp(-|\xi - \lambda_k|), \quad \xi \in \mathbb{R},$$

*is positive definite.*

Note that in spite of the complex valued character of the individual terms, we assume that the whole sum is real for all  $x \in \mathbb{R}$ . For instance we can consider a real combination of trigonometric functions

$$f(x) = \sum_{k=1}^n (a_k \cos(\lambda_k x) + b_k \sin(\lambda_k x)), \quad a_k, b_k \in \mathbb{R}.$$

For the proof we have to simply recall the Fourier transform

$$\int_{-\infty}^{\infty} e^{-i\xi x} \frac{dx}{1+x^2} = \pi e^{-|\xi|},$$

and apply Bochner's Theorem to the positive measure

$$\frac{\sum_{k=1}^n c_k \exp(i\lambda_k x)}{1+x^2} dx.$$

Again, see [1] for details.

A major simplification is available in the case the dependence of the entries  $\lambda_k$  over  $\mathbb{Q}$  is known. For instance, due to Kronecker's observation that the set  $\{(\exp(i\lambda_k x))_i : x \in \mathbb{R}\}$  is dense in the unit torus  $\mathbb{T}^n$  (i.e.,  $\mathbb{T} := \{z \in \mathbb{C} : |z| = 1\}$ ) whenever  $\lambda_1, \dots, \lambda_n$  are independent over  $\mathbb{Q}$ , we can reduce the above criterion to positivity of a polynomial as follows.

**Corollary 14.3.** *Assume that  $\mu_1, \dots, \mu_m$  are independent over  $\mathbb{Q}$  and that  $\lambda_k = \gamma_{k1}\mu_1 + \dots + \gamma_{km}\mu_m$ , where  $\gamma_k = (\gamma_{kj}) \in \mathbb{N}^m$ . Then  $\sum_{k=1}^n c_k \exp(i\lambda_k x) \geq 0$  for all  $x \in \mathbb{R}$ , if and only if*

$$\sum_{k=1}^n c_k \zeta^{\gamma_k} \geq 0, \quad \forall \zeta = (\zeta_1, \dots, \zeta_m) \in \mathbb{T}^m. \quad (14.13)$$

Observe that writing  $\zeta_j = x_j + iy_j$ ,  $j = 1, \dots, n$ , (14.13) reduces to checking whether some real polynomial  $h \in \mathbb{R}[\mathbf{x}, \mathbf{y}]$  is nonnegative on the compact set  $\mathbf{K} := \{(\mathbf{x}, \mathbf{y}) : x_k^2 + y_k^2 = 1, k = 1, \dots, n\}$ . Again, violation of nonnegativity of  $h$  on  $\mathbf{K}$  can be certified by solving finitely many semidefinite relaxations of the hierarchy defined in [32].

### 14.3.5 Continuous Piecewise Polynomials

Continuous splines do not need an introduction for their many applications. Although their positivity falls into the framework of the present survey, a systematic study of sums of squares decompositions of continuous piecewise polynomial functions has just begun with the note [41]. We indicate below the basic ideas, leaving to the reader the task to extend them in various applied directions.

Let  $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \dots \cup \mathcal{A}_k \subset \mathbb{R}^n$  be a finite simplicial complex in euclidean space  $\mathbb{R}^n$ , that is, every  $\mathcal{A}_i \subset \mathbb{R}^n$  is a  $d_i$ -dimensional simplex such that  $\mathcal{A}_i \cap \mathcal{A}_j$  is a face of both  $\mathcal{A}_i$  and  $\mathcal{A}_j$  for all  $i, j = 1, \dots, k$ . It is of pure-dimension if all  $\mathcal{A}_i$ 's have same dimension. The algebra  $C^0(\mathcal{A})$  of continuous functions on  $\mathcal{A}$  which are polynomials when restricted to every  $\mathcal{A}_j$  was fully described algebraically and combinatorially in Billera [5]. Let  $v_1, \dots, v_m$  be the vertices of the complex  $\mathcal{A}$ . The *tent function*, or *Courant function*, associated to the vertex  $v_j$  is the unique piecewise linear function  $T_j$  satisfying  $T_j(v_k) = 0$  if  $k \neq j$  and  $T_j(v_j) = 1$ . In [5] Billera proved that these functions generate the algebra  $C^0(\mathcal{A})$  modulo the algebraic relations:

$$\sum_{j=1}^m T_j = 1, \quad T_i T_j T_k = 0, \quad T_\alpha T_\beta = 0,$$

where  $(i, j, k)$  are mutually distinct indices and  $(\alpha, \beta), \alpha \neq \beta$ , is not an edge of  $\Delta$ . Besides the expected weighted sums of squares decomposition of a positive element of  $C^0(\Delta)$  derived from general Positivstellensatze in the polynomial ring with generators  $T_j$ , the following special case was singled out recently.

**Theorem 14.4 (Plaumann [41]).** *Let  $\Delta \subset \mathbb{R}^n$  be a simplicial complex of pure dimension 1 with  $m$  vertices and  $e$  edges. Let  $T_1, \dots, T_m \in C^0(\Delta)$  be the tent functions of  $\Delta$ . A function  $f \in C^0(\Delta)$  is non-negative on  $\Delta$  if and only if there exists sums of squares  $S, S_{ij} \in \Sigma C^0(\Delta)^2$  with the property*

$$f = S + \sum_{i \neq j} S_{ij} T_i T_j. \quad (14.14)$$

*More precisely, one can choose  $S$  and  $S_{ij}$  of degree at most  $\deg(f) + 8(e - 1) + 2$ , where  $S$  is a sum of at most  $2e$  squares and each  $S_{ij}$  is a sum of two squares.*

Note that, due to the relations satisfied by the tent functions, the sum in the statement ranges only over the edges of  $\Delta$ . We refer to [41] for the proof.

## 14.4 Optimization of Semi-algebraic Functions

In most examples considered in the previous sections, one has shown that positivity of the semi-algebraic functions involved (and sometimes positivity of some of non semi-algebraic functions) reduces to positivity of polynomials defined in a lifted space. And so, as one may expect, the apparatus available for polynomial optimization also works for these functions.

For illustration purposes, let  $\mathcal{A}$  be the algebra of functions considered in Sect. 14.3.2, let  $\mathbf{K}, \mathbf{S} \subset \mathbb{R}^d$  be as in (14.5) and (14.9) respectively, and consider the global optimization problem:

$$\mathbf{P}: \quad f^* := \min_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{S}\}. \quad (14.15)$$

**Proposition 14.9.** *With  $\mathbf{K}$  in (14.5) compact and with  $\mathbf{S}$  as in (14.9)-(14.10), let  $f \in \mathcal{A}$  have the b.s.a.l.*

$$\Psi_f = \{(\mathbf{x}, z_f) : \mathbf{x} \in \mathbf{K}; q_\ell(\mathbf{x}, \mathbf{z}) = 0, \quad \ell = 1, \dots, t_f; z_j \geq 0, \quad j \in J_f\},$$

*where  $\mathbf{z} = (z_1, \dots, z_{m_f})$  and  $J \subseteq \{1, \dots, m_f\}$ . Then  $\mathbf{P}$  is also the polynomial optimization problem:*

$$f^* = \min_{\mathbf{x}, \mathbf{y}, \mathbf{z}} \{z_{m_f} : (\mathbf{x}, \mathbf{y}, \mathbf{z}) \in \Omega\}$$

where  $\Omega \subset \mathbb{R}^{n+t+m_f}$  is the basic semi-algebraic set

$$\begin{aligned}\Omega = \{(\mathbf{x}, \mathbf{y}, \mathbf{z}) : & g_j(\mathbf{x}) \geq 0, \quad j = 1, \dots, m \\ & u_k(\mathbf{x}, \mathbf{y}) = 0, \quad k = 1, \dots, r \\ & q_\ell(\mathbf{x}, \mathbf{z}) = 0, \quad \ell = 1, \dots, t_f \\ & y_k, z_j \geq 0, \quad k \in J, j \in J_f\}\end{aligned}\tag{14.16}$$

This follows from the fact that for every  $(\mathbf{x}, \mathbf{y}, \mathbf{z}) \in \Omega$ ,  $z_{m_f}$  is exactly equal to  $f(\mathbf{x})$  when  $\mathbf{x} \in \mathbf{S}$ . Then one may apply the moment-sos approach developed in e.g. [30, 32] and build up a hierarchy of semidefinite programs  $(\mathbf{Q}_i)_i$ . If in the definition of  $\Omega$ , one adds the (redundant) archimedean quadratic constraint  $M - \|(\mathbf{x}, \mathbf{y}, \mathbf{z})\|^2 \geq 0$ , then the sequence of optimal values  $(\rho_i)_i$  associated with the hierarchy  $(\mathbf{Q}_i)_i$  converges to the global optimum  $f^*$ . In addition, if a certain rank condition is satisfied at an optimal solution of some semidefinite relaxation  $\mathbf{Q}_k$ , then  $\rho_k = f^*$  (i.e. finite convergence takes place) and one may extract global minimizers. For more details see [30, 32].

#### 14.4.1 Examples

Consider the following two examples of  $\mathbf{P}$  in (14.15).

*Example 14.9.* Let  $d = 2$  and  $\mathbf{P} : f^* = \max_{\mathbf{x}} \{x_1 |x_1 - 2x_2| : \|\mathbf{x}\|^2 = 1\}$ . Hence

$$\Omega = \{(\mathbf{x}, z) : \|\mathbf{x}\|^2 - 1 = 0; z^2 - (x_1 - 2x_2)^2 = 0; z \geq 0\}.$$

One does not need to add the archimedean quadratic constraint  $M - \|(\mathbf{x}, \mathbf{z})\|^2 \geq 0$  for some  $M$  large enough, because in view of the constraint  $1 = \|\mathbf{x}\|^2$ , the polynomial  $(\mathbf{x}, \mathbf{z}) \mapsto M - \|(\mathbf{x}, \mathbf{z})\|^2$  is in the quadratic module generated by the polynomials that define  $\Omega$ . At the second semidefinite relaxation  $\mathbf{Q}_2$ , the rank condition is satisfied<sup>3</sup> so that  $f^* = \rho_2 = 1.6180$  and one may extract the global minimizer  $(\mathbf{x}^*, z^*) = (0.8507, -0.5257, 1.9021)$ . (In fact, one even has  $\rho_1 = f^*$ .)

*Example 14.10.* Consider  $d = 2$  and  $\mathbf{P} : f^* = \max_{\mathbf{x}} \{f(\mathbf{x}) : \mathbf{x} \in \mathbf{K} = [-1, 1]^2\}$ , where  $f$  is the b.s.a. function of Example 14.5 with  $\mathbf{x} \mapsto p_1(\mathbf{x}) := 2x_1$ ,  $\mathbf{x} \mapsto p_2(\mathbf{x}) := x_2$ ,  $\mathbf{x} \mapsto \ell(\mathbf{x}) := 0.1 + x_1^2 + x_2^2$ . So with  $\mathbf{x} \mapsto g_i(\mathbf{x}) = 1 - x_i^2$ ,  $i = 1, 2$ ,  $\Omega \subset \mathbb{R}^8$  is the set

---

<sup>3</sup>The semidefinite relaxation was solved with the GloptiPoly software [17] dedicated to solving the generalized problem of moments as defined in [32].

$$\begin{aligned} \Omega = \{(\mathbf{x}, \mathbf{z}) \in \mathbb{R}^8 : & \quad \mathbf{x} \in \mathbf{K}; z_i^2 - p_i(\mathbf{x})^2 = 0; \quad z_i \geq 0; \quad i = 1, 2; \\ & z_3^{2q+1} - y_1^{2q+1} - y_2^{2q+1} = 0; \\ & z_4^2 - \ell(\mathbf{x}) - z_3 = 0; \quad z_4 \geq 0; \\ & z_5^2 - (a(\mathbf{x}) - y_4)^2 = 0; \quad z_5 \geq 0; \\ & 2z_6 - (a(\mathbf{x}) + y_4) + z_5 = 0 \}. \end{aligned}$$

Again, one does not need to add the archimedean quadratic constraint  $M - \|(\mathbf{x}, \mathbf{z})\|^2 \geq 0$  because with  $g_i(\mathbf{x}) = 1 - x_i^2$ ,  $i = 1, 2$ , the polynomial  $(\mathbf{x}, \mathbf{z}) \mapsto M - \|(\mathbf{x}, \mathbf{z})\|^2$  is in the quadratic module generated by the polynomials that define  $\Omega$ . The semidefinite relaxation  $\mathbf{Q}_3$  (i.e. with moments up to order 6) provides the global minimum  $f^* = 0.4942$  and one may extract the global minimizer  $\mathbf{x} = (0, 0)$ .

#### 14.4.2 Sparsity

In the description of  $\Omega$  in (14.16), a *lifting*  $\mathbf{y}^\ell \in \mathbb{R}^{n_\ell}$  is associated with each non-polynomial function  $h_\ell \in \mathcal{A}$  that appears in the description of  $\mathbf{P}$ , as well as a lifting  $\mathbf{z} \in \mathbb{R}^{m_f}$  for  $f$  (if non-polynomial). This can be rapidly penalizing. However it is worth noting that there is no mixing between the lifting variables  $\mathbf{z}$  and  $\mathbf{y}^\ell$ , as well as between the lifting variables  $\mathbf{y}^\ell$  and  $\mathbf{y}^k$ ,  $k \neq \ell$ . The coupling of all these variables is through the variables  $\mathbf{x}$ . Hence there is an obvious sparsity pattern if the set of all variables  $(\mathbf{x}, \mathbf{y}, \mathbf{z})$  is written as the union (with overlaps)

$$(\mathbf{x}, \mathbf{y}, \mathbf{z}) = (\mathbf{x}) \cup (\mathbf{x}, \mathbf{z}) \cup (\mathbf{x}, \mathbf{y}^1) \cup \cdots \cup (\mathbf{x}, \mathbf{y}^s)$$

and this sparsity pattern obviously satisfies the co-called *running intersection property*. Indeed, for each  $1 \leq k < s$ ,

$$(\mathbf{x}, \mathbf{y}^{k+1}) \bigcap \left( (\mathbf{x}) \cup (\mathbf{x}, \mathbf{z}) \cup \bigcup_{i=1}^k (\mathbf{x}, \mathbf{y}^i) \right) \subseteq (\mathbf{x}).$$

See e.g. [31]. This is good news because with such a sparsity pattern one may use the sparse semidefinite relaxations defined in [50], whose associated sequence of optimal values was still proved to converge to  $f^*$  in [31]. Hence if on the one hand lifting is penalizing, on the other hand the special structure of the lifted polynomial optimization problem permits to use specialized “sparse” relaxations, which (partly) compensates the increase in the number of variables. For more details on sparse semidefinite relaxations, the interested reader is referred to e.g. [31] and [50].

## 14.5 Stability of Differential and Difference Equations with Delays

In this final section, and as another application of positivity for non-polynomial functions, we consider the *stability* of trajectories that are solutions of difference or differential equations with delays in the argument, which is an important issue in control. From the vast literature we have isolated two relevant stability criteria and for complete details the interested reader is advised to consult the monographs [11, 13, 35].

Let  $A_k \in M(d, d; \mathbb{R})$ ,  $1 \leq k \leq n$ , be real square matrices of size  $d \times d$ , let  $r_k \geq 0$  be non-negative real numbers, and let  $t \mapsto \mathbf{x}(t) \in \mathbb{R}^d$  be the unknown function subject to the difference relation

$$\mathbf{x}(t) = \sum_{k=1}^n A_k \mathbf{x}(t - r_k), \quad t \in [-r, \infty), \quad (14.17)$$

where  $r = \max_k r_k$ . Knowing the values of  $\mathbf{x}(t)$  for  $t \in [-r, 0]$  determines  $\mathbf{x}(t)$  on the whole positive semi-axis. The following criterion was established in the mid seventies by Henry and Hale, see [2]: *If*

$$\sup_{\lambda} \left\{ \operatorname{Re} \lambda : \det \left[ I - \sum_{k=1}^n A_k \exp(-\lambda r_k) \right] = 0 \right\} \leq 0,$$

*then the delay system (14.17) is stable.* After developing the determinant and introducing the vectors  $\gamma_j \in \mathbb{N}^d$ , one can write

$$\det \left[ I - \sum_{k=1}^n A_k \exp(-\lambda r_k) \right] = 1 + \sum_{j=1}^N a_j \exp(-\lambda \gamma_j \cdot r) = h(\lambda, a, r),$$

where  $a_j$  are real coefficients and  $r = (r_1, \dots, r_d)$ .

The dependence of the zero set  $V(h) = \{\lambda : h(\lambda, a, r) = 0\}$  on the parameters  $a, r$  is quite delicate and makes the subject of a detailed analysis in the article [2]. The following major observation goes back to Henry [14]. Recall that  $\mathbb{T} \subset \mathbb{C}$  denotes the unit torus  $\{z \in \mathbb{C} : |z| = 1\}$ .

**Theorem 14.5.** *Denote  $\rho = \operatorname{Re} \lambda$  and assume that the components of the vector  $r$  are independent over  $\mathbb{Q}$ . Then  $\rho \in \overline{V(h)}$  if and only if there is  $\zeta \in \mathbb{T}^d$  with the property*

$$1 + \sum_{j=1}^N a_j e^{-\rho \gamma_j \cdot r} \zeta^{\gamma_j} = 0. \quad (14.18)$$

Observe that with the change of variables  $\mathbf{y} = (y_1, \dots, y_d) \in (0, \infty)^d$  where  $y_k := e^{-\rho r_k}$  for all  $k$ , the function in (14.18) becomes the polynomial  $1 + \sum_{j=1}^N a_j y^{\gamma_j} \zeta^{\gamma_j}$ . It is however the continuous parameter  $\rho = \frac{1}{r_k} \log \frac{1}{y_k}$  which counts in the stability of the original system together with the implicit non-polynomial constraints  $y_k^{r_\ell} = y_\ell^{r_k}$ ,  $1 \leq k, \ell \leq d$ .

### 14.5.1 The Case Where the $r_k$ 's Are Commensurable

If for all  $k$ ,  $r_k = t_k/s_k$  for some  $t_k, s_k \in \mathbb{N}$ , then with  $s$  being the least common multiple of the  $s_k$ 's and doing the change of variable  $u := e^{-\rho/s}$ , (14.18) reduces to check whether

$$\min_{u, \zeta} \{u : 1 + p(u, \zeta) = 0; \zeta \in \mathbb{T}; u \geq 0\} \geq 1,$$

with  $(u, \zeta) \mapsto p(u, \zeta) := \sum_{j=1}^N a_j u^{\alpha_j} \zeta^{\gamma_j} \in \mathbb{R}[u, \zeta]$ , for some  $(\alpha_j)_j \subset \mathbb{N}^d$ . Equivalently, writing  $\zeta_j := x_j + iy_j$ ,  $j = 1, \dots, d$ , as well as  $p = p_1 + ip_2$  with  $p_1, p_2 \in \mathbb{R}[u, \mathbf{x}, \mathbf{y}]$ , checking (14.18) reduces to checking whether  $u^* \geq 1$ , where

$$u^* := \min_{u, \mathbf{x}, \mathbf{y}} \left\{ u : 1 + p_1(u, \mathbf{x}, \mathbf{y}) = 0; p_2(u, \mathbf{x}, \mathbf{y}) = 0; u \geq 0; x_j^2 + y_j^2 = 1, j = 1, \dots, d \right\}.$$

Again, one may then apply the hierarchy of semidefinite relaxations defined in [32] to obtain a nondecreasing sequence of lower bounds  $(u_k^*)_k$  such that  $u_k^* \uparrow u^*$  as  $k \rightarrow \infty$ . If  $u_k^* \geq 1$  for some  $k$ , one may stop and certify stability.

### 14.5.2 The General Case

For the general case we are less ambitious and consider fixed arbitrary values of  $\rho$ . Again for fixed  $\rho := \Re \lambda$ , checking (14.18) reduces to check whether some complex polynomial  $p \in \mathbb{C}[\zeta]$  has a zero on  $\mathbb{T}^d$ . And so a first step towards understanding condition (14.18) is provided by the following observation, essentially due to Stengle [3].

**Proposition 14.10.** *Let  $p \in \mathbb{C}[z_1, \dots, z_d]$  be a complex polynomial. Then  $p$  has no zero on  $\mathbb{T}^d$  if and only if there are finitely many polynomials  $f_i, g_j \in \mathbb{C}[z_1, \dots, z_d]$  and  $h_k \in \mathbb{R}[x_1, \dots, x_d, y_1, \dots, y_d]$  such that*

$$\left( \sum_i |f_i(\mathbf{z})|^2 \right) |p(\mathbf{z})|^2 = 1 + \sum_j |g_j|^2 + \sum_k (1 - |z_k|^2) h_k(\mathbf{x}, \mathbf{y}) \quad (14.19)$$

(where  $\mathbf{z} = \mathbf{x} + i\mathbf{y}$ ).

We now indicate how to implement an algorithm to check Proposition 14.10. Basically, and modulo the ideal that defines the torus, one may work with real sums of squares. Indeed:

**Proposition 14.11.** *Let  $p \in \mathbb{C}[\mathbf{z}, \bar{\mathbf{z}}]$  be a complex polynomial and let  $z_k = x_k + iy_k$ , for every  $k = 1, \dots, n$  and let  $i := \sqrt{-1}$ .*

(a) *Let  $p \in \mathbb{C}[\mathbf{z}, \bar{\mathbf{z}}]$ . Then  $|p(\mathbf{z}, \bar{\mathbf{z}})|^2 = |q(\mathbf{z})|^2 + \sum_{k=1}^n (1 - z_k \bar{z}_k) h_k(\mathbf{x}, \mathbf{y})$  for some  $q \in \mathbb{C}[\mathbf{z}], h_k \in \mathbb{R}[\mathbf{x}, \mathbf{y}]$ .*

(b) *Let  $p \in \mathbb{R}[\mathbf{x}, \mathbf{y}]$ . Then  $p(\mathbf{x}, \mathbf{y})^2 = |q(\mathbf{z})|^2 + \sum_{k=1}^n (1 - z_k \bar{z}_k) h_k(\mathbf{x}, \mathbf{y})$  for some  $q \in \mathbb{C}[\mathbf{z}], h_k \in \mathbb{R}[\mathbf{x}, \mathbf{y}], k = 1, \dots, n$ .*

*Proof.* For notational simplicity we prove it in the univariate case.

(a) Let  $N$  be the largest power of  $\bar{z}$  which appears in the monomials  $z^k \bar{z}^j$  of  $p$  and use the ideal  $\langle z\bar{z} - 1 \rangle$  to write  $p(z, \bar{z}) = \bar{z}^N q(z) + (1 - z\bar{z})\theta(z, \bar{z})$  for some  $q \in \mathbb{C}[z]$  and  $\theta \in \mathbb{C}[z, \bar{z}]$ . Then multiplying by  $p(z, \bar{z})$  yields

$$\begin{aligned} |p(z, \bar{z})|^2 &= (\bar{z}\bar{z})^N |q(z)|^2 + (1 - z\bar{z})w(z, \bar{z}) \quad [\text{with } w(z, \bar{z}) = \overline{w(z, \bar{z})}] \\ &= |q(z)|^2 + (1 - z\bar{z}) \left[ w(z, \bar{z}) - (1 + z\bar{z} + \dots + (z\bar{z})^{N-1}) |q(z)|^2 \right] \\ &= |q(z)|^2 + (1 - z\bar{z})\Theta(z, \bar{z}) \quad [\text{with } \Theta(z, \bar{z}) = \overline{\Theta(z, \bar{z})}] \\ &= |q(z)|^2 + (1 - z\bar{z})h(x, y) \end{aligned}$$

for some  $h \in \mathbb{R}[x, y]$ .

(b) Let  $x := (z + \bar{z})/2$  and  $y := (z - \bar{z})/(2i)$  so that  $p(x, y) = p((z + \bar{z})/2, (z - \bar{z})/2i) = P(z, \bar{z})$  with  $P \in \mathbb{C}[z, \bar{z}]$ . However,  $P(z, \bar{z}) = \overline{P(z, \bar{z})}$  and so  $P(z, \bar{z})^2 = |P(z, \bar{z})|^2$ . Applying (a) yields the desired result.  $\square$

Therefore in view of Proposition 14.11, checking whether (14.19) holds reduces to check whether

$$\sigma(\mathbf{x}, \mathbf{y})|p(\mathbf{z})|^2 = 1 + \psi(\mathbf{x}, \mathbf{y}) + \sum_{k=1}^n (1 - x_k^2 - y_k^2) h_k(\mathbf{x}, \mathbf{y})$$

for some polynomials  $\sigma, \psi \in \Sigma \mathbb{R}[\mathbf{x}, \mathbf{y}]^2$  and some  $(h_k) \subset \mathbb{R}[\mathbf{x}, \mathbf{y}]$ . And, with an a priori fixed degree bound on  $\sigma, \psi$  and  $h_k$ , this can be checked by solving a semidefinite program.

One step further, when dealing with differential equations with delays in the argument:

$$\dot{\mathbf{x}}(t) = A_0 \mathbf{x}(t) + \sum_{k=1}^n A_k \mathbf{x}(t - \gamma_k \cdot r),$$

one encounters a very similar stability criterion. As before,  $A_k \in M(d, d; \mathbb{R})$ ,  $1 \leq k \leq n$  are real matrices of size  $d \times d$ ,  $r = (r_1, \dots, r_N), r_k \geq 0$  are non-negative real numbers,  $\gamma_k \in \mathbb{N}$ , and  $\mathbf{x}(t) \in \mathbb{R}^d$ . The characteristic function

$$\lambda \mapsto f(\lambda, r, A) := \det \left[ \lambda I - A_0 - \sum_k A_k e^{-\lambda y_k \cdot r} \right],$$

its zero set  $V(f)$  and the projection of  $V(f)$  on the real axis are the main objectives. In practice one seeks stability of the solution  $\mathbf{x}(t)$  in terms of the matrices  $A_0, \dots, A_n$ , for all possible values  $r \in \mathbb{R}^N$ . This boils down to the following condition

$$\det \left[ iy - A_0 - \sum_{k=1}^n A_k \zeta^{y_k} \right] \neq 0, \quad y \in \mathbb{R}, y \neq 0, \zeta \in \mathbb{T}^d, \quad (14.20)$$

and  $\operatorname{Re}\sigma(A_0 + \dots + A_n) < 0$  [12].

Again, Proposition 14.10 offers a first step towards casting the above relation into an algebraic certificate.

## 14.6 Guide to the Literature

A chapter of a handbook obviously requires some orientation through the existing literature. By its inherent interdisciplinary contents, the present chapter relies on quite a few distinct mathematical ideas, and sources. A rough classification of them, reflected in the bibliography, follows.

For the algebraic and geometric aspects of *polynomial inequalities* we recommend the monographs [3, 40] the collection of articles [43] and the landmark note by Stengle [47]. The *moment problem* for power monomials, trigonometric polynomials or more general elementary functions is treated in the classic monographs [1, 24, 27] and in the articles [6, 20, 23, 26, 45]. The *functional analytic approach* to moment problems (via linear functionals and the spectral theorem) is exposed in the classic monographs [25, 44] and the articles [16, 28, 42, 46]. The *positivity of non-polynomial functions* is treated algebraically in the works [5, 7–9, 34, 36, 41] and more analytically in [18, 19, 51].

The recent spectacular *applications of real algebraic geometry to optimization* problems are treated in the monograph [32], the collections of articles [38, 43] and in [4, 6, 29–31, 33, 39, 50]. The related *computer algorithms* package can be found in [17].

The *stability of delay systems* is a big enterprise in itself. The monographs [11, 13, 21, 35, 37] and the articles [2, 10, 12, 14, 15, 22, 37, 48, 49] offer a partial view on this topic, as related to the subject of our chapter.

**Acknowledgements** The authors thank Daniel Plaumann for a careful reading and constructive comments on an earlier version of the manuscript. Partially supported by the National Science Foundation DMS 10-01071, U.S.A.

## References

1. Akhiezer, N.I.: The classical moment problem and some related questions in analysis. Translated from the Russian by N. Kemmer, Hafner Publishing Co., New York (1965)
2. Avellar, C.E., Hale, J.K.: On the zeros of exponential polynomials. *J. Math. Analysis Appl.* **73**, 434–452 (1980)
3. Bochnack, J., Coste, M., Roy, M.-F.: Real Algebraic Geometry. Ergeb. Math. Grenzgebiete (3rd series), vol. 36. Springer, Berlin-New York (1998)
4. Bertsimas, D., Popescu, I.: Optimal inequalities in probability theory: a convex optimization approach, *SIAM J. Optim.* **15**(3), 780–804 (2005)
5. Billera, L.J.: The algebra of continuous piecewise polynomials. *Adv. Math.* **76**, 170–183 (1989)
6. Bojanov, B., Petrov, P.: Gaussian interval quadrature formulae for Tchebycheff systems. *SIAM J. Numer. Anal.* **43**, 787–795 (2005)
7. Burgdorf, S., Scheiderer, C., Schweighofer, M.: Pure states, nonnegative polynomials, and sums of squares. To appear in *Comm. Math. Helvet.*
8. Delzell, C.N.: A continuous, constructive solution to Hilbert’s 17-th problem. *Invent. Math.* **76**, 365–384 (1984)
9. Delzell, C.N.: Continuous, piecewise polynomial functions which solve Hilbert’s 17-th problem. *J. reine Angew. Math.* **440**, 157–173 (1993)
10. Fliess, M., Mounier, H.: Quelques propriétés structurelles des systèmes linéaires à retards constants. *C.R. Acad. Sci. Paris, I-319*, 289–294 (1994)
11. Gu, K., Kharitonov, V.L., Chen, J.: Stability and robust stability of time-delay systems. Birkhauser: Boston (2003)
12. Hale, J.K., Infante, E.F., Tsen, F.S.-P.: Stability in linear delay equations *J. Math. Anal. Appl.* **105**, 533–555 (1985)
13. Hale, J.K., Verduyn Lunel, S.M.: Introduction to Functional Differential Equations. AMS, vol. 99. Springer, New York (1993)
14. Henry, D.: Linear autonomous neutral functional differential equations. *J. Differential Equations* **15**, 106–128 (1974)
15. Hertz, D., Jury, E.I., Zeheb, E.: Root exclusion from complex polydomains and some of its applications. *Automatica* **23**, 399–404 (1987)
16. Helton, J.W. Putinar, M.: Positive Polynomials in Scalar and Matrix Variables, the Spectral Theorem and Optimization. In: Operator Theory, Structured Matrices, and Dilations, pp. 229–306. Theta, Bucharest (2007)
17. Henrion, D., Lasserre, J.B., Löfberg, J.: GloptiPoly 3: moments, optimization and semidefinite programming. *Optim. Methods and Software* **24**, 761–779 (2009)
18. Isii, K.: The extrema of probability determined by generalized moments. I. Bounded random variables. *Ann. Inst. Statist. Math.* **12**, 119–134 (1960)
19. Isii, K.: Inequalities of the types of Chebyshev and Cramer-Rao and mathematical programming. *Ann. Inst. Statist. Math.* **16**, 277–293 (1964)
20. Kakutani, S.: Ein Beweis des Satzes von M. Eidelheit über konvexe Mengen. *Proc. Imp. Acad. Tokyo* **13**, 93–94 (1937)
21. Kamen, E.W.: Lectures on Algebraic System Theory: Linear Systems over Rings. NASA Contractor Report 3016 (1978)
22. Kamen, E.W.: On the relationship between zero criteria for two-variable polynomials and asymptotic stability of delay differential equations. *IEEE Trans. Automat. Contr.* **AC-25**, 983–984 (1980)
23. Karlin, S., Studden, W.J.: Optimal experimental designs. *Ann. Math. Statist.* **37**, 783–815 (1966)
24. Karlin, S., Studden, W.J.: Tchebycheff systems: With applications in analysis and statistics. Pure and Applied Mathematics, vol. XV. Interscience Publishers John Wiley & Sons, New York-London-Sydney (1966)

25. Köthe, G.: Topological Vector Spaces. I. Springer, Berlin (1969)
26. Krein, M.G.: The ideas of P.L. Tchebycheff and A.A. Markov in the theory of limiting values of integrals and their further development. (Russian) *Uspehi Matem. Nauk (N.S.)* **6**, 3–120 (1951)
27. Krein, M.G., Nudelman, A.: The Markov moment problem and extremal problems. Ideas and problems of P.L. Tchebycheff and A.A. Markov and their further development. Translated from the Russian by D. Louvish, Translations of Mathematical Monographs, vol. 50. American Mathematical Society, Providence, R.I. (1977)
28. Krein, M.G., Rutman, M.A.: Linear operators leaving invariant a cone in a Banach space (in Russian). *Uspehi Mat. Nauk* **23**, 3–95 (1948)
29. Lasserre, J.B.: Optimisation globale et théorie des moments. *C. R. Acad. Sci. Paris* **331** Série 1, 929–934 (2000)
30. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. *SIAM J. Optim.* **11**, 796–817 (2001)
31. Lasserre, J.B.: Convergent SDP-relaxations in polynomial optimization with sparsity. *SIAM J. Optim.* **17**, 822–843 (2006)
32. Lasserre, J.B.: Moments, Positive Polynomials and Their Applications. Imperial College Press, London (2009)
33. Lasserre, J.B., Putinar, M.: Positivity and optimization for semi-algebraic functions. *SIAM J. Optimization* **10**, 3364–3383 (2010)
34. Marshall, M., Netzer, T.: Positivstellensätze for real function algebras. *Math. Zeit.* (2010)
35. Michiels, W., Niculescu, S.-I.: Stability and stabilization of time-delay systems. An eigenvalue based approach. SIAM: Philadelphia (2007)
36. Monnier, J.-P.: Anneaux d'holomorphie et Positivstellensatz archimédien. *Manuscripta Math.* **97**, 269–302 (1998)
37. Niculescu, S.-I.: Stability and hyperbolicity of linear systems with delayed state: A matrix pencil approach. *IMA J. Math. Contr. Information* **15**, 331–347 (1998)
38. Henrion, D., Garulli, A. (eds.): Positive Polynomials in Control. Lecture Notes in Control and Information Sciences, Springer, Berlin, Heidelberg (2005)
39. Parrilo, P.A.: Semidefinite programming relaxations for semialgebraic problems. *Math. Program. Ser. B* **96**, 293–320 (2003)
40. Prestel, A., Delzell, C.: Positive polynomials. Springer, Berlin (2001)
41. Plaumann, D.: Positivity of piecewise polynomials. Preprint (2010)
42. Putinar, M.: Positive polynomials on compact semi-algebraic sets. *Indiana Univ. Math. J.* **42**, 969–984 (1993)
43. Putinar, M., Sullivant, S. (eds.): Emerging Applications of Algebraic Geometry. IMA Series in Applied Mathematics, Springer, Berlin (2009)
44. Riesz, F., Sz.-Nagy, B.: Functional analysis. In: Transl. from the 2nd French ed. by Leo F. Boron. Reprint of the 1955 orig. publ. by Ungar Publ. Co., Dover Books on Advanced Mathematics, Dover Publications, Inc., New York (1990)
45. Riesz, M.: Sur le problème des moments. *Troisième Note. Ark. Mat. Fys.* **16**, 1–52 (1923)
46. Schmüdgen, K.: The K-moment problem for compact semi-algebraic sets. *Math. Ann.* **289** 203–206 (1991)
47. Stengle, G.: A Nullstellensatz and a Positivstellensatz in semi-algebraic geometry. *Math. Ann.* **207**, 87–97 (1974)
48. Thowsen, A.: An analytical stability test for a class of time-delay systems. *IEEE Trans. Automat. Contr.* **AC-25** 735–736 (1981)
49. Toker, O., Özbay, H.: Complexity issues in robust stability of linear delay-differential systems. *Math., Contr., Signals, Syst.* **9**, 386–400 (1996)
50. Waki, H., Kim, S., Kojima, M., Maramatsu, M.: Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity. *SIAM J. Optim.* **17**, 218–242 (2006)
51. Zielke, R.: Discontinuous Tchebycheff systems. Lecture Notes in Mathematics 707, Springer, Berlin (1979)

## **Part II**

# **Algorithms**

# Chapter 15

## Self-Regular Interior-Point Methods for Semidefinite Optimization

Maziar Salahi and Tamás Terlaky

### 15.1 Introduction

From the 1990s, Semidefinite Optimization (SDO), which yields a generalization of linear optimization (LO) problems, has become one of the most active area both in mathematical programming and real life applications. More and more difficult problems arising from practice could be modeled as SDO problems. Nevertheless, when we carefully review the literature, the first paper which discusses an SDO problem can be traced back to the 1960s [5], almost four decades ago. One of the reasons that SDO was out of interest for such a long time was the lack of robust and efficient algorithms for solving SDO problems. This situation has changed since the 1990s. Alizadeh [2] and Nesterov and Nemirovskii [12] independently developed the first interior-point methods (IPMs) for SDO. Alizadeh [2] applied Ye's potential reduction idea to SDO and showed how variants of dual IPMs could be extended to SDO. Almost at the same time, in their milestone book [12], Nesterov and Nemirovskii proved that IPMs are able to solve general conic optimization problems, in particular SDO problems, in polynomial time. After that, notable successes are achieved in the theory and practice of SDO. Nowadays, algorithms for SDO has been already matured and one can solve large scale problems even on a desktop PC. Another reason that SDO was out of interest for a long time is the limited realization of its importance in various applications. In [3], Alizadeh showed how SDO could be used in combinatorial optimization. One of the landmark work was done by Goemans and Williamson [8]. Using SDO they proposed a randomized

---

M. Salahi (✉)

Department of Applied Mathematics, Faculty of Mathematical Sciences,  
University of Guilan, Rasht, Iran  
e-mail: [salahim@guilan.ac.ir](mailto:salahim@guilan.ac.ir)

T. Terlaky

Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA  
e-mail: [terlaky@lehigh.edu](mailto:terlaky@lehigh.edu)

0.878-approximation algorithm for the famous NP hard MAX-CUT problem. Later this bound was further tightened [11]. In the past two decades, the applications of SDO were expanded to new areas, including control theory, probability theory, statistics, signal processing and structural design [21].

As in the case of LO, there is an intriguing fact about IPMs for SDO. Although their theoretical complexity is worse, large neighborhood algorithms perform better in practice than small neighborhood algorithms. Many efforts were spent to bridge this gap. In [17], Peng, Roos and Terlaky established a new paradigm based on the class of the so-called self-regular functions. Under their new paradigm, the complexity bound for large neighborhood IPMs may come as close as a  $\log n$  factor close to the best known iteration complexity bounds of small neighborhood IPMs. In this chapter we discuss this framework with its complexity analysis. Furthermore, motivated by the self-regularity, Roos et al., Peyghami, Bai et al., [6, 7, 18, 22] introduced new kernel functions that still allow to derive similar iteration complexity bounds as that of self-regular IPMs.

We consider the following form of SDO

$$\begin{aligned} \min \quad & C \bullet X \\ A_i \bullet X = b_i, \quad & i = 1, \dots, m, \\ X \geq 0, \quad & \end{aligned} \tag{15.1}$$

where  $X \in S^n$  ( $S^n$  is the space of  $n$  by  $n$  symmetric matrices),  $X \geq 0$  means that matrix  $X$  is positive semidefinite, and  $A \bullet B = \text{Tr}(A^T B)$ . The dual SDO problem is given by

$$\begin{aligned} \max \quad & b^T y \\ \sum_{i=1}^m y_i A_i \leq C, \quad & \end{aligned} \tag{15.2}$$

where  $\sum_{i=1}^m y_i A_i \leq C$  means that the symmetric matrix  $C - \sum_{i=1}^m y_i A_i$  is positive semidefinite. To be more precise, we should write  $\inf C \bullet X$  in the primal problem and  $\sup b^T y$  in the dual problem, because in general the infimum of the primal (and the supremum of the dual problem, respectively) might not be attained, thus an optimal solution might not exist. If an optimal solution exists, we say that the corresponding problem is solvable.

Let us denote the primal feasible set by  $\mathcal{F}_P$  i.e.,

$$\mathcal{F}_P = \{X \geq 0 \mid A_i \bullet X = b_i, i = 1, \dots, m\}$$

and its interior by  $\mathcal{F}_P^0$  i.e.,

$$\mathcal{F}_P^0 = \{X > 0 \mid A_i \bullet X = b_i, i = 1, \dots, m\}.$$

Analogously for the dual problem the feasible set is denoted by  $\mathcal{F}_D$  i.e.,

$$\mathcal{F}_D = \left\{ S \geq 0 \mid \sum_{i=1}^m y_i A_i + S = C \right\},$$

and its interior by  $\mathcal{F}_D^0$  i.e.,

$$\mathcal{F}_D^0 = \left\{ S > 0 \mid \sum_{i=1}^m y_i A_i + S = C \right\}.$$

We may assume without loss of generality that the matrices  $A_i$ ,  $i = 1, \dots, m$  are linearly independent. Thus a given  $S \in \mathcal{F}_D$  uniquely determines  $y \in \mathbb{R}^m$ , so we may use  $S \in \mathcal{F}_D$  and  $(y, S) \in \mathcal{F}_D$  interchangeably.

**Theorem 15.1 (Weak Duality).** *If  $X \in \mathcal{F}_P$  and  $(y, S) \in \mathcal{F}_D$ , then*

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

The value  $C \bullet X - b^T y = X \bullet S$  is referred to as the duality gap.

**Corollary 15.1.** *If we have  $X^* \in \mathcal{F}_P$  and  $(y^*, S^*) \in \mathcal{F}_D$  with  $C \bullet X^* = b^T y^*$ , then  $X^*$  is a primal optimal and  $(y^*, S^*)$  is a dual optimal solution with zero duality gap.*

In case of LO, the duality gap for optimal solutions, by the strong duality theorem [19] is zero. Unlike the LO case, the strong duality for SDO requires an extra assumption to be hold.

**Theorem 15.2 (Strong Duality).** *Assume that both of the following two conditions hold:*

- (i) *Problem (15.1) is strictly feasible, i.e., there exists an  $X \in \mathcal{F}_P^0$ .*
- (ii) *Problem (15.2) is strictly feasible, i.e., there exists  $S \in \mathbb{S}^n$ ,  $(y, S) \in \mathcal{F}_D^0$ .*

*Then there exist a pair of optimal solutions  $X^* \in \mathcal{F}_P$  and  $(y^*, S^*) \in \mathcal{F}_D$  satisfying  $C \bullet X^* = b^T y^*$ , i.e., optimal solutions with zero duality gap.*

*Moreover, if condition (i) holds and problem (15.1) is bounded below, then the duality gap is zero, and the dual problem is solvable. If condition (ii) holds and problem (15.2) is bounded above, then the duality gap is zero and the primal problem is solvable. In both cases the optimal values of the primal and dual problems are equal.*

The optimality conditions for an SDO problem can be outlined as follows:

$$\begin{aligned} A_i \bullet X = b_i, \quad X \geq 0, \quad i = 1, \dots, m, \\ \sum_{i=1}^m y_i A_i + S = C, \quad S \geq 0, \\ X \bullet S = 0. \end{aligned} \tag{15.3}$$

The first two sets of equations represent primal and dual feasibility, respectively. The last set of nonlinear equations is called the *complementarity condition*. Obviously,  $X \bullet S = 0$  can be replaced by the equivalent condition  $XS = 0$ .

As in the LO case, IPMs for SDO follow the central path approximately to reach an optimal solution. The central path for SDO is defined by the solution set  $\{X(\mu), y(\mu), S(\mu) : \mu > 0\}$  of the system

$$\begin{aligned} A_i \bullet X = b_i, \quad X > 0, i = 1, \dots, m, \\ \sum_{i=1}^m y_i A_i + S = C, \quad S > 0, \\ XS = \mu I, \end{aligned} \tag{15.4}$$

where  $I$  denotes the  $n \times n$  identity matrix and  $\mu \geq 0$  is the so-called barrier parameter. For every  $\mu > 0$ , system (15.4) has a unique solution [23]. The basic idea of IPMs is to follow this path as  $\mu$  goes to zero to find an approximate optimal solution. Suppose the point  $(X, y, S)$  is strictly feasible. By applying Newton's method to (15.4), we have the following linear system of equations

$$A_i \bullet \Delta X = 0, \quad i = 1, \dots, m, \tag{15.5}$$

$$\sum_{i=1}^m \Delta y_i A_i + \Delta S = 0, \tag{15.6}$$

$$X \Delta S + \Delta X S = \mu I - XS. \tag{15.7}$$

A crucial observation is that the computed  $\Delta X$  from this system might not be symmetric. Over the years, people suggested many strategies to deal with this issue. Alizadeh, Haeberly and Overton (AHO) [4] suggested to symmetrize both sides of (15.7). Another possible alternative is to employ a similarity transformation  $P(\cdot)P^{-1}$  on both sides of (15.7). This strategy was first investigated by Monteiro [20] for  $P = X^{-\frac{1}{2}}$  and  $P = S^{\frac{1}{2}}$ . It turned out that the resulting directions by this approach could be seen as two special cases of the class of directions introduced earlier by Kojima et al. [10]. At the same time, another motivation led Helmberg et al. [9] to the direction given by  $P = S^{\frac{1}{2}}$ . The search directions given by  $P = X^{-\frac{1}{2}}$  and  $P = S^{\frac{1}{2}}$  are usually referred to as the H..K..M directions, respectively. Another very popular direction was introduced by Nesterov and Todd [13, 14] in their attempt to generalize primal-dual IPMs beyond SDO. In [24], based on Monteiro's idea, Zhang generalized all the approaches to a unified scheme parameterized by a nonsingular scaling matrix  $P$ . This family of search directions is referred to as the Monteiro–Zhang (MZ) family of search directions, where the last set of equations in (15.4) is replaced by

$$H_P(XS) := \frac{1}{2} (PXS P^{-1} + P^{-T} SXP^T) = \mu I. \tag{15.8}$$

Thus (15.7) becomes

$$P(XS + X\Delta S + \Delta XS)P^{-1} + P^{-T}(SX + S\Delta X + \Delta SX)P^T = 2\mu I. \quad (15.9)$$

Different choices of matrix  $P$  in (15.8) and subsequently in (15.9) results in different symmetrization schemes. We use the Nesterov–Todd (NT) symmetrization scheme [13–16] which is defined by  $P = W^{\frac{1}{2}}$ , where

$$W := X^{\frac{1}{2}}(X^{\frac{1}{2}}SX^{\frac{1}{2}})^{-\frac{1}{2}}X^{\frac{1}{2}} = S^{-\frac{1}{2}}(S^{\frac{1}{2}}XS^{\frac{1}{2}})^{\frac{1}{2}}S^{-\frac{1}{2}}.$$

Now let  $D = W^{\frac{1}{2}}$ , then obviously  $D$  is symmetric and positive definite and can be used to scale the matrices  $X$  and  $S$  and to define the symmetric and positive definite matrix  $V$  as follows:

$$V := \frac{1}{\sqrt{\mu}}D^{-1}XD^{-1} = \frac{1}{\sqrt{\mu}}DSD, \quad (15.10)$$

This notation enables us to express the centrality condition as  $V = I$ . Let us further define

$$\bar{A}_i := DA_iD, \quad i = 1, \dots, m,$$

$$D_X := \frac{1}{\sqrt{\mu}}D^{-1}\Delta XD^{-1}, \quad D_S = \frac{1}{\sqrt{\mu}}D\Delta S D.$$

Now the NT search direction can be written as the unique solution of the following system

$$\begin{aligned} \text{Tr}(\bar{A}_i D_X) &= 0, \quad i = 1, \dots, m \\ D_S + \sum_{i=1}^m \Delta y_i \bar{A}_i &= 0 \\ D_X + D_S &= V^{-1} - V. \end{aligned} \quad (15.11)$$

In the sequel the right hand side of the third set of equations in (15.11) is replaced by the negative gradient of a SR proximity function,<sup>1</sup> and thus we present SR-IPMs using NT scaling [15–17].

<sup>1</sup>As we see the right hand side of the last equation in (15.11) is the negative gradient of the classical logarithmic barrier function  $\psi(V)$  (see Definition 3 on p. 9), where

$$\psi(t) = \frac{t^2 - 1}{2} - \log t.$$

## 15.2 Self-Regular IPMs

Let us now introduce the class of *Self-Regular* (SR) functions and present some of their properties for later use. Then we describe a family of SR-IPMs and highlight the major differences between the SR-approach and the classical approaches.

### 15.2.1 SR Functions

The class of SR functions was introduced by Peng et al. [17] as follows.

**Definition 15.1.** A twice continuously differentiable function  $\psi(t) : (0, \infty) \rightarrow \mathcal{R}$  is SR if it satisfies the following two conditions:

SR.1: The function  $\psi(t)$  is strictly convex with respect to  $t > 0$  and vanishes at its global minimal point  $t = 1$ , i.e.,  $\psi(1) = \psi'(1) = 0$ . Further, there exist positive constants  $v_2 \geq v_1 > 0$  and  $p \geq 1$ ,  $q \geq 1$  such that

$$v_1(t^{p-1} + t^{-1-q}) \leq \psi''(t) \leq v_2(t^{p-1} + t^{-1-q}), \quad \forall t \in (0, \infty); \quad (15.12)$$

SR.2: For any  $t_1, t_2 > 0$ ,

$$\psi(t_1^r t_2^{1-r}) \leq r\psi(t_1) + (1-r)\psi(t_2), \quad \forall r \in [0, 1]. \quad (15.13)$$

If  $\psi(t)$  is SR, then parameter  $q$  is called the *barrier degree* and parameter  $p$  is called the *growth degree* of the SR function  $\psi(t)$ .

There are two popular families of SR functions. The first one is given by

$$\Upsilon_{p,q}(t) = \frac{t^{p+1} - 1}{p(p+1)} + \frac{t^{1-q} - 1}{q(q-1)} + \frac{p-q}{pq}(t-1), \quad p \geq 1, q > 1, \quad (15.14)$$

with  $v_1 = v_2 = 1$ . One can easily derive this family by integrating twice the inequalities in (15.12). The second family, which is a slight modification of (15.14), is given by

$$\Gamma_{p,q}(t) = \frac{t^{p+1} - 1}{p+1} + \frac{t^{1-q} - 1}{q-1}, \quad p \geq 1, q > 1, \quad (15.15)$$

with  $v_1 = \min(p, q)$  and  $v_2 = \max(p, q)$ . For  $p, q = 1$  in both cases the classical logarithmic barrier function

$$\Gamma_{11}(t) = \Upsilon_{11}(t) = \frac{t^2 - 1}{2} - \log t$$

is obtained.

Now we present some of the fundamental properties of SR functions. To avoid too much technical details we present only the proofs of some lemmas and theorems. The interested reader can consult [17] for more details.

**Proposition 15.1.** *If the functions  $\psi_1$  and  $\psi_2$  are SR functions, then any conic combination  $\beta_1\psi_1 + \beta_2\psi_2$ , where  $\beta_1, \beta_2 \geq 0$ ,  $\beta_1 + \beta_2 > 0$  is also SR.*

Let us denote by  $\mathcal{Q}_1$  and  $\mathcal{Q}_2$  the set of twice continuously differentiable functions that satisfy conditions SR.1 and SR.2, respectively.

**Lemma 15.1.** *If  $\psi(t) = \psi(t^{-1})$  and  $\psi(t) \in \mathcal{Q}_1$ , then  $\psi(t)$  is SR.*

The following lemma gives a different characterization of condition SR.2. It is known as the exponential-convexity and has been used for further derivation of new kernel functions in [6, 7].

**Lemma 15.2.** *A function  $\psi(t)$  belongs to  $\mathcal{Q}_2$  if and only if the function  $\psi(\exp(\zeta))$  is convex in  $\zeta$  or, equivalently if  $\psi'(t) + t\psi''(t) \geq 0$  for  $t > 0$ .*

*Proof.* We know that  $\psi(\exp(\zeta))$  is convex if and only if for any  $\zeta_1, \zeta_2 \in R$ , the following inequality holds:

$$\psi(\exp(r\zeta_1 + (1-r)\zeta_2)) \leq r\psi(\exp(\zeta_1)) + (1-r)\psi(\exp(\zeta_2)), \quad \forall r \in [0, 1].$$

Now let  $t_1 = \exp(\zeta_1)$  and  $t_2 = \exp(\zeta_2)$ . Obviously  $t_1, t_2 \in (0, +\infty)$  and the previous inequality is equivalent to

$$\psi(t_1^r t_2^{1-r}) \leq r\psi(t_1) + (1-r)\psi(t_2), \quad r \in [0, 1].$$

Thus  $\psi(t) \in \mathcal{Q}_2$ . Moreover, since  $\psi(\exp(\zeta))$  is convex, thus its second derivative is nonnegative i.e.,

$$\exp(2\zeta)\psi''(\exp(\zeta)) + \exp(\zeta)\psi'(\exp(\zeta)) \geq 0.$$

Now if we substitute  $t = \exp(\zeta)$ , then we have  $t\psi'(t) + t^2\psi''(t) \geq 0$  or  $\psi'(t) + t\psi''(t) \geq 0$  for  $t > 0$ .  $\square$

**Lemma 15.3.** *Suppose that  $\psi(t) \in \mathcal{Q}_2$ . Then for any  $\alpha \in \mathcal{R}$ ,  $\psi(t^\alpha) \in \mathcal{Q}_2$ .*

The following lemma is used to prove some fundamental results in the sequel which are key elements in the analysis of our new algorithms.

**Lemma 15.4.** *Suppose that  $\psi(t) \in \mathcal{Q}_1$ . Then*

$$\frac{(t-1)^2}{2} \leq \frac{\psi(t)}{\nu_1},$$

$$\frac{t^{p+1}-1}{p(p+1)} - \frac{t-1}{p} \leq \frac{\psi(t)}{\nu_1},$$

$$\frac{t^{1-q}-1}{q(q-1)} - \frac{t-1}{q} \leq \frac{\psi(t)}{\nu_1},$$

$$\psi(t) \leq \frac{\psi'(t)^2}{2\nu_1}.$$

The next lemma gives some lower bounds for  $|\psi'(t)|$  that can also be interpreted as the barrier behavior of the function  $\psi'(t)$  and  $\psi(t)$ . These bounds are used in the proof of some further results.

**Lemma 15.5.** *Suppose that  $\psi(t) \in \mathcal{Q}_1$ . Then*

$$|\psi'(t)| \geq \frac{\nu_1(t^{-q}-1)}{q}, \quad \forall t < 1$$

and

$$|\psi'(t)| \geq \frac{\nu_1(t^p-1)}{p}, \quad \forall t > 1.$$

*Proof.* From condition SR.1, whenever  $t < 1$ , one has

$$\begin{aligned} \psi'(t) &= \int_1^t \psi''(\zeta) d\zeta \leq \nu_1 \int_1^t (\zeta^{p-1} + \zeta^{-1-q}) d\zeta \leq \nu_1 \\ &\int_1^t \zeta^{-1-q} d\zeta = \frac{\nu_1}{q}(1-t^{-q}). \end{aligned}$$

This gives the first inequality of the lemma. One can prove the second inequality analogously.  $\square$

### 15.2.2 SR-Proximity Measures

In this section first we introduce a class of SR proximity functions that are measuring the distance of the current iterate from the central path. Then we present some of their fundamental properties. Finally, we extend this definition to the space of symmetric positive semidefinite matrices.

**Definition 15.2.** A SR-proximity measure  $\Psi(v) : \mathcal{R}_{++}^n \rightarrow \mathcal{R}_+$  is defined by

$$\Psi(v) := \sum_{i=1}^n \psi(v_i), \tag{15.16}$$

where  $\psi(\cdot)$  is SR, and the subscript ‘++’ denotes the interior of the nonnegative orthant.

For notational convenience we use

$$\sigma = \|\nabla \Psi(v)\|. \quad (15.17)$$

The following result establishes a relationship between the proximity measure and its gradient norm, and gives a lower bound for the smallest, and an upper bound for the largest component of a given vector  $v$ . This result is used frequently in the analysis of SR-IPMs.

**Proposition 15.2.** *Let the SR proximity measure  $\Psi(v)$  be defined by (15.16). Then*

$$\begin{aligned} \Psi(v) &\leq \frac{\sigma^2}{2v_1}, \\ v_{\min} &\geq \left(1 + \frac{q\sigma}{v_1}\right)^{-\frac{1}{q}} \\ v_{\max} &\leq \left(1 + \frac{p\sigma}{v_1}\right)^{\frac{1}{p}}. \end{aligned}$$

*Proof.* The first statement follows from the last statement of Lemma 15.4. To prove the second inequality, we first note that it holds trivially if  $v_{\min} \geq 1$ . If  $v_{\min} < 1$ , then from Lemma 15.5 one has

$$\sigma \geq |\psi'(v_{\min})| \geq \frac{v_1}{q} \left( \frac{1}{v_{\min}^q} - 1 \right),$$

which implies the second statement of the proposition. The last statement can be proved analogously.  $\square$

The following theorem gives lower and upper bounds for the smallest and largest components of the vector  $v$ , respectively, when one uses (15.15) with  $p = 1$  as the kernel function in the SR-Proximity measure given by (15.16). These bounds are much tighter than the bounds in the previous proposition, and lead to better results in our development.

**Theorem 15.3.** *Let (15.15) be the kernel function with  $p = 1$  and  $q \geq 1$  in the definition of the SR-proximity measure given by (15.16). Then*

$$\begin{aligned} v_{\min} &\geq (1 + \sigma)^{-\frac{1}{q}}, \\ v_{\max} &\leq (1 + \sigma). \end{aligned}$$

*Proof.* The statements are trivial when  $v_{\min} \geq 1$  and  $v_{\max} < 1$ . Now let us assume that  $v_{\min} < 1$ . Then from (15.17) one has

$$\sigma = \|v - v^{-q}\| \geq |v_i^{-q} - v_i|, \quad \forall i = 1, \dots, n.$$

This implies

$$\sigma \geq \frac{1}{v_{\min}^q} - v_{\min} \geq \frac{1}{v_{\min}^q} - 1,$$

which implies the first statement of the theorem. Now let us consider  $v_{\max} > 1$ . Then analogous to the previous case one has

$$\sigma \geq v_{\max} - \frac{1}{v_{\max}^q} \geq v_{\max} - 1,$$

which completes the proof of the second statement and finally the proof of the theorem.  $\square$

Now we are ready to define SR proximity functions for matrix variables.

**Definition 15.3.** Suppose that the matrix  $X$  is diagonalizable with eigenvalue decomposition

$$X = Q_X^{-1} \operatorname{diag}(\lambda_1(X), \lambda_2(X), \dots, \lambda_n(X)) Q_X,$$

where  $Q_X$  is nonsingular, and let  $\psi(t)$  be a function from  $\mathcal{R}$  into itself. The function  $\psi(X)$  is defined by

$$\psi(X) = Q_X^{-1} \operatorname{diag}(\psi(\lambda_1(X)), \psi(\lambda_2(X)), \dots, \psi(\lambda_n(X))) Q_X. \quad (15.18)$$

In particular, if  $X$  is symmetric then  $Q_X$  can be chosen to be orthogonal.

**Definition 15.4.** A matrix function  $\psi(X)$  defined by (15.18) is SR in  $S_{++}^n$ , where  $S_{++}^n$  denotes the space of symmetric positive definite matrices, if the kernel function  $\psi(t)$  is SR.

Having defined the matrix function  $\psi(X)$  allows us to define an SR proximity function  $\Psi(X)$  for SDO as follows:

$$\Psi(X) := \sum_{i=1}^n \psi(\lambda_i(X)) = \operatorname{Tr}(\psi(X)). \quad (15.19)$$

**Proposition 15.3.** Suppose that  $X$  is symmetric positive definite and that the functions  $\psi(X)$  and  $\Psi(X)$  are defined by (15.18) and (15.19), respectively. If  $\psi(X)$  is a SR function on  $S_{++}^n$ , then the following statements hold:

C.1. The matrix function  $\Psi(X)$  is strictly convex with respect to  $X > 0$  and vanishes at its global minimal point  $X = I$ , i.e.,  $\Psi(I) = 0$ ,  $\psi(I) = \psi'(I) = 0_{n \times n}$ . Further, there exist two positive constants  $v_2 > v_1 > 0$  and  $p, q \geq 1$ , such that

$$v_1(X^{p-1} + X^{-1-q}) \leq \psi''(X) \leq v_2(X^{p-1} + X^{-1-q}), \quad \forall X > 0. \quad (15.20)$$

C.2. For any  $X_1, X_2 > 0$ ,

$$\Psi\left([X_1^{\frac{1}{2}} X_2 X_1^{\frac{1}{2}}]^{\frac{1}{2}}\right) \leq \frac{1}{2}(\Psi(X_1) + \Psi(X_2)). \quad (15.21)$$

*Proof.* See Proposition 5.2.6 of [17].  $\square$

The two statements in the proposition can be viewed as transparent extensions of the SR conditions in the LO case.

### 15.2.3 SR Primal-Dual IPMs for SDO

As we mentioned at the end of Sect. 15.1, in SR-IPMs the right hand side of third set of equations in (15.11) is replaced by the negative gradient of the SR proximity measure as follows:

$$\begin{aligned} \text{Tr}(\bar{A}_i D_X) &= 0, \quad i = 1, \dots, m \\ D_S + \sum_{i=1}^m \Delta y_i \bar{A}_i &= 0 \\ D_X + D_S &= -\psi'(V), \end{aligned} \quad (15.22)$$

where  $\psi'(V) = Q_V^{-1} \text{diag}(\psi'(\lambda_i(V))) Q_V$ .

In this section we present IPMs based on this system and present their iteration complexity results. As these IPMs are defined by utilizing SR proximity functions, they are referred to as SR-IPMs.

Due to the orthogonality of  $\Delta X$  and  $\Delta S$  in (15.22), we have  $\text{Tr}(D_X D_S) = \text{Tr}(D_S D_X) = 0$ . Here we outline the scheme of SR-IPMs for SDO.

Analogous to the LO case, to measure the distance of the iterates from the central path, the proximity measure suggested for SDO is defined by

$$\Phi(X, S, \mu) := \Psi(V) = \text{Tr}(\psi(V)),$$

where  $\psi(V)$  is SR. In the sequel we progress to characterize SR proximities for SDO. For notational brevity, we also define

$$\sigma^2 = \text{Tr}(\psi'(V)^2) = \|\psi'(V)\|^2.$$

**Proposition 15.4.** *If function  $\Psi(\cdot)$  is SR, then we have*

$$\Psi(V) \leq \frac{\sigma^2}{2\nu_1},$$

$$\lambda_{\min}(V) \geq \left(1 + \frac{q\sigma}{\nu_1}\right)^{-\frac{1}{q}},$$

$$\lambda_{\max}(V) \leq \left(1 + \frac{p\sigma}{\nu_1}\right)^{\frac{1}{p}}.$$

Now we are ready to present the scheme of SR Primal-Dual IPMs

---

### SR Primal-Dual IPM for SDO

---

#### Inputs

A proximity parameter  $\tau \geq \nu_1^{-1}$ ;  
 an accuracy parameter  $\epsilon > 0$ ;  
 a fixed barrier update parameter  $\theta \in (0, 1)$ ;  
 $(X^0, S^0) \in \mathcal{F}_P^0 \times \mathcal{F}_D^0$  and  $\mu^0 = 1$  such that  $\Psi(V^0) \leq \tau$ ;  
 $X := X^0; S = S^0; \mu = \mu^0$ ;

**while**  $n\mu \geq \epsilon$  **do**

$\mu := (1 - \theta)\mu$ ;

**while**  $\Psi(V) \geq \tau$  **do**

Solve system (15.22) for  $\Delta X, \Delta y, \Delta S$ ;

Determine a step size  $\alpha$ ;

Set  $X = X + \alpha \Delta X$

$y = y + \alpha \Delta y$

$S = S + \alpha \Delta S$

**end**

**end**

---

In the sequel the decrease of the proximity measure after each step is estimated. To do so, let us denote by  $V_+$  the scaled matrix defined by (15.10), where matrices  $X$  and  $S$  are replaced by  $X_+ = X + \alpha \Delta X$  and  $S_+ = S + \alpha \Delta S$ , respectively. Obviously  $V_+^2$  is a unitary matrix similar to  $X_+^{\frac{1}{2}} S_+ X_+^{\frac{1}{2}}$ , and thus to  $(V + \alpha D_X)^{\frac{1}{2}} (V + \alpha D_S) (V + \alpha D_X)^{\frac{1}{2}}$  as well. Consequently, the eigenvalues of  $V_+$  are exactly the same as of

$$\tilde{V}_+ = \left( (V + \alpha D_X)^{\frac{1}{2}} (V + \alpha D_S) (V + \alpha D_X)^{\frac{1}{2}} \right)^{\frac{1}{2}}. \quad (15.23)$$

Since the proximity measure after one step is  $\Psi(V_+)$ , then from (15.23) we have

$$\Psi(V_+) = \Psi(\tilde{V}_+).$$

Our goal is to give an estimate of the decrease of the proximity measure after one step, namely

$$f(\alpha) := \Psi(V_+) - \Psi(V) = \Psi(\tilde{V}_+) - \Psi(V).$$

Obviously from condition C.2. we have

$$f(\alpha) \leq \frac{1}{2} (\Psi(V + \alpha D_X) + \Psi(V + \alpha D_S)) - \Psi(V) := f_1(\alpha).$$

Therefore, it is sufficient to give an estimate of the decrease of  $f_1(\alpha)$ . To do so, we have

$$f'_1(\alpha) = \frac{1}{2} \text{Tr}(\psi'(V + \alpha D_X) D_X + \psi'(V + \alpha D_S) D_S),$$

and

$$f''_1(\alpha) = \frac{1}{2} \frac{d^2}{d\alpha^2} \text{Tr}(\psi(V + \alpha D_X) + \psi(V + \alpha D_S)).$$

**Lemma 15.6.** *If the step size  $\alpha \leq \bar{\alpha} = \sigma^{-1} \left(1 + \frac{q\sigma}{\nu_1}\right)^{-\frac{1}{q}}$ , then*

$$f''_1(\alpha) \leq \frac{\nu_2 \sigma^2}{2} \left( (\lambda_{\max}(V) + \alpha\sigma)^{p-1} + (\lambda_{\min}(V) - \alpha\sigma)^{-q-1} \right).$$

*Proof.* Lemma 5.4.2 on p. 120 of [17]. □

Then the decrease of the proximity function can be estimated as follows.

**Theorem 15.4.** *We have*

$$f(\alpha) \leq -\frac{\nu_5 \nu_1^{\frac{q-1}{2q}}}{4} \Psi(V)^{\frac{q-1}{2q}}.$$

*Proof.* We have

$$f'(0) = f'_1(0) = -\frac{\sigma^2}{2}.$$

Therefore, since  $f(0) = f_1(0) = 0$ , from Lemma 15.6 we have

$$\begin{aligned} f(\alpha) &\leq f_1(\alpha) \leq f_2(\alpha) := \\ &-\frac{\sigma^2 \alpha}{2} + \frac{\nu_2 \sigma^2}{2} \int_0^\alpha \int_0^\zeta \left( (\lambda_{\max}(V) + \alpha\sigma)^{p-1} + (\lambda_{\min}(V) - \alpha\sigma)^{-q-1} \right) d\eta d\zeta. \end{aligned}$$

Obviously,  $f_2(\alpha)$  is convex and twice differentiable for all  $\alpha \in [0, \bar{\alpha}]$ . It is also easy to see that  $f_2(\alpha)$  is decreasing at zero and goes to infinity as  $\alpha \rightarrow \bar{\alpha}$ . Let  $\alpha^*$  be the point at which  $f_2(\alpha)$  attains its global minimal value, i.e.,

$$\alpha^* = \arg \min_{\alpha \in [0, \bar{\alpha}]} f_2(\alpha).$$

For this  $\alpha^*$ , by means of Lemma 1.3.3 of [17], we obtain

$$f(\alpha^*) \leq f_2(\alpha^*) \leq \frac{1}{2} f'_2(0) \alpha^* = \frac{1}{2} f'(0) \alpha^*.$$

Moreover one can get

$$\alpha^* \geq \nu_5 \sigma^{-\frac{q+1}{q}},$$

where  $\nu_5$  is a constant. Therefore we have the desired result.  $\square$

**Corollary 15.2.** *If  $\psi(t) = \Gamma_{p,q}(t)$ , then*

$$f(\alpha) \leq -\min\left\{\frac{1}{12p+4}, \frac{1}{24q+16}\right\} \Psi(V)^{\frac{q-1}{2q}}.$$

Now we can derive a worst case iteration complexity bound for SR-IPMs.

**Lemma 15.7.** *Let  $\Phi(X, S, \mu) \leq \tau$  and  $\tau \geq \nu_1^{-1}$ . Then after an update of  $\mu$ , no more than*

$$\left\lceil \frac{4q\nu_1^{-\frac{q-1}{2q}}}{\nu_5} (\psi_0(\theta, \tau, n))^{\frac{q+1}{2q}} \right\rceil$$

iterations are needed to recenter, where

$$\psi_0(\theta, \tau, n) = \frac{\nu_2 \tau}{\nu_1 \theta_1^{\frac{p+1}{2}}} + \nu_2 \Upsilon'_{p,q}(\theta^{-\frac{1}{2}}) \sqrt{\frac{2n\tau}{\nu_1 \theta_1}} + n \nu_2 \Upsilon_{p,q}(\theta_1^{-\frac{1}{2}}), \quad \theta_1 = 1 - \theta.$$

In the special case when  $\psi(t) = \Gamma_{p,q}(t)$  (with  $\nu_1 = \nu_2 = 1$ ), this bound is

$$\left\lceil \frac{8q \max\{3p+1, 6q+4\}}{q+1} (\psi_0(\theta, \tau, n))^{\frac{q+1}{2q}} \right\rceil.$$

**Theorem 15.5.** *If  $\tau \geq \nu_1^{-1}$ , the total number of iterations required by a SR primal-dual IPM is no more than*

$$\left\lceil \frac{4q\nu_1^{-\frac{q-1}{2q}}}{\nu_5} (\psi_0(\theta, \tau, n))^{\frac{q+1}{2q}} \right\rceil \left\lceil \frac{1}{\theta} \log \frac{n}{\epsilon} \right\rceil$$

which obviously for the specific choice in the previous lemma reduces to

$$\left\lceil \frac{8q \max\{3p+1, 6q+4\}}{q+1} (\psi_0(\theta, \tau, n))^{\frac{q+1}{2q}} \right\rceil \left\lceil \frac{1}{\theta} \log \frac{n}{\epsilon} \right\rceil.$$

Moreover, by choosing  $\theta \in (0, 1)$  and suitable  $p, q \geq 1$ , the complexity of large update SR-IPMs is  $O\left(n^{\frac{q+1}{2q}} \log \frac{n}{\epsilon}\right)$ , while for a small update algorithm with  $\theta = O(\frac{1}{\sqrt{n}})$  is  $O(\sqrt{n} \log \frac{n}{\epsilon})$ . Further, if we set  $p$  a constant and  $q = \log n$ , the large update algorithm has an  $O\left(\sqrt{n} \log n \log \frac{n}{\epsilon}\right)$  worst case iteration complexity.

### 15.3 Non SR Kernel Functions for SDO

Right after introducing the framework of self-regularity, several non SR kernel functions have been introduced that are utilized as kernel functions in IPMs for SDO. In specific cases they also admit the same worst case iteration complexity that has been achieved by using SR functions. Not surprising that some new analysis tools were needed to achieve those results. In the sequel we present some of the non-SR kernel functions with the corresponding complexity bounds.

- Wang et al. [22] have used the following family of kernel functions to define the proximity measure in their algorithm:

$$\psi(t) = t - 1 + \frac{t^{1-q} - 1}{q-1}, \quad t > 0,$$

where  $q > 1$ . This kernel function does not belong to the class of SR functions because the growth term is only linear. Consequently a large part of the analysis tools developed in [17] do not longer apply. It has the following fundamental properties that are playing crucial role in proving polynomial iteration complexity of IPMs using this kernel function that has the following properties:

- (i)  $\psi'(1) = \psi(1) = 0$ .
- (ii)  $\psi''(t) > 0$ , for all  $t > 0$ .
- (iii)  $\lim_{t \rightarrow 0} \psi(t) = \lim_{t \rightarrow \infty} \psi(t) = \infty$ .

The worst case iteration complexity bounds for small and large-update algorithms are  $O(q^2 \sqrt{n} \log \frac{n}{\epsilon})$  and  $O(qn \log \frac{n}{\epsilon})$ , respectively.

- El Ghami et al. [6] have given a more general algorithm based on eligible kernel functions.

A function  $\psi : (0, \infty) \rightarrow [0, \infty)$  is called a kernel function if it is three time differentiable and has properties (i) to (iii). Clearly, (i) and (ii) say that  $\psi(t)$  is a nonnegative strictly convex function which has its global minimum at  $t = 1$ , with  $\psi(1) = 0$ . Note that with these properties  $\psi(t)$  is completely determined by its second derivative

$$\psi(t) = \int_1^t \int_1^\xi \psi''(\zeta) d\zeta d\xi,$$

because its value and first derivative is fixed to zero. Further, property (iii), imply that  $\psi(t)$  has the barrier property.

For eligible kernel functions we require the following properties:

$$\begin{aligned} t\psi''(t) + \psi'(t) &> 0, \quad t < 1 \\ \psi'''(t) &< 0, \quad t > 0 \\ 2\psi''(t)^2 - \psi'(t)\psi'''(t) &> 0, \quad t < 1 \\ \psi''(t)\psi'(\beta t) - \beta\psi''(t)\psi'(t) &> 0, \quad t > 1, \beta > 1. \end{aligned}$$

Using these properties, the analysis of [7] give an  $O(\sqrt{n} \log n \log \frac{n}{\epsilon})$ , and  $O(\sqrt{n} \log \frac{n}{\epsilon})$  iterations complexity bound for the related large and small-update IPMs, respectively.

- El Ghami et al., [7] have introduced the following kernel functions that are not SR:

$$\psi(t) = \frac{t^{1+p} - 1}{1 + p} + \frac{e^{\sigma(1-t)} - 1}{\sigma}, \quad p \in [0, 1], \sigma \geq 1, t \geq 0.$$

Obviously, if  $p < 1$ , then these functions are not strongly convex, and hence also not SR. The induced barrier functions differ from the existing barrier functions in the sense that they are finite at the boundary of the feasible region. For this kernel function  $\lim_{t \rightarrow \infty} \psi(t) = \infty$ , while  $\lim_{t \rightarrow 0} \psi(t) = \frac{e^\sigma - 1}{\sigma} - \frac{1}{1+p}$ . This means that if either  $X$  or  $S$  approaches the boundary of the feasible region, then the proximity measure  $\Psi(V)$  converges to a finite value, depending on the value of  $\sigma$ . This kernel function has another feature which dominates the behavior if  $t$  approaches  $\infty$ . Analogous to the SR function  $\psi(1) = \psi'(1) = 0$ , and  $\psi''(t) > 0$ , showing that  $\psi(t)$  is strictly convex and has global minimum at  $t = 1$ . Using these fundamental properties and several technical lemmas Roos et al., [7] proved for  $p = 1$  and  $\sigma = O(\log n)$  that  $O(\sqrt{n} \log n \log \frac{n}{\epsilon})$  is the iteration complexity of this specific kernel function based large-update IPMs for SDO.

- Recently, Peyghami [18] introduced the following kernel functions

$$\psi_{pq}(t) = \frac{t^2 - 1}{2} - \int_1^t e^{p(x^{-q}-1)} dx,$$

where  $p, q \geq 1$ . One can easily check that this class of kernel functions do not belong to the class of SR functions. Using the properties of these kernel functions and following the same track of reasoning as of [22], the related primal-dual IPM for SDO has an

$$O\left(pq \sqrt{n} \left(1 + \frac{1}{p} \log n\right)^{1+\frac{1}{q}} \log \frac{n}{\epsilon}\right)$$

iteration complexity bound. Obviously by choosing  $p = O(\log n)$  and  $q = 1$ , we achieve the best  $O(\sqrt{n} \log n \log \frac{n}{\epsilon})$  iteration complexity result for large update SDO algorithms.

Most recently, Terlaky and Yang [20], by generalizing the Ai-Zhang [1] direction, introduced a new class of large neighborhood IPMs for SDO problems. An important ingredient of their algorithm is the decomposition of the classical Newton direction to two individual directions: one of them reduces the duality gap and the other one keeps the iterate away from the boundary of the positive semidefinite cone. Furthermore, using the Monteiro–Zhang symmetrization scheme [23], they proved that the algorithm stops with an  $\epsilon$ -approximate solution after at most  $O(\eta \sqrt{\kappa_\infty n} \log \frac{X^0 \cdot S^0}{\epsilon})$  iterations, where  $\eta$  is the neighborhood parameter and  $\kappa_\infty$  is associated with the scaling matrix  $P$  in the symmetrization scheme. In case of the NT direction, the iteration complexity bound of the algorithm is  $O(\sqrt{n} \log \frac{X^0 \cdot S^0}{\epsilon})$ , which is exactly the same as for classical small-update methods.

**Second Order Cone Optimization (SOCO):** Another class of conic optimization problems is the class of SOCO problems. IPMs for SOCO are proved to be polynomial for various scaling techniques [23]. Moreover, SR-IPMs [17] for SOCO enjoy analogous iteration complexity bounds as for SDO problems. Among them, NT scaling based IPMs have the lowest iteration complexity bound for large update IPMs. For further details the reader is referred to [16]. We also mention that the above mentioned non-SR kernel function based IPMs were not analyzed for SOCO problems. To analyze those algorithms for SOCO remains to the interested reader.

## 15.4 Concluding Remarks

In this chapter an extensive overview of SR-IPMs for SDO based on the Nesterov–Todd direction is given. Furthermore, several none SR kernel functions are reviewed as well. IPMs based on theses kernel functions are enjoying similar iteration complexity bounds as SR-IPMs, but different analysis needed as they have different properties.

**Acknowledgements** Research of the second author was supported by a start-up grant of Lehigh University, and by the Hungarian National Development Agency and the European Union within the frame of the project TAMOP 4.2.2-08/1-2008-0021 at the Széchenyi István University entitled “Simulation and Optimization - basic research in numerical mathematics”.

## References

1. Ai, W., Zhang, S.: An  $O(\sqrt{n}L)$  iteration primal-dual path-following method, based on wide neighborhood and large updates, for monotone LCP. *SIAM J. on Optimization* **16**, 400–417 (2005)
2. Alizadeh, F.: Combinatorial optimization with interior-point methods and semi-definite matrices. Ph.D. Thesis, Computer Science Department, University of Minnesota, Minneapolis, MN, USA (1991)

3. Alizadeh, F.: Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. on Optimization* **5**, 13–51 (1995)
4. Alizadeh, F., Haeberly, J.A., Overton, M.L.: Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results. *SIAM J. on Optimization* **8**, 746–768 (1998)
5. Bellman, R., Fan, K.: On systems of linear inequalities in Hermitian matrix variables. In: Klee, V.L. (ed.) *Convexity, Proceedings of Symposia in Pure Mathematics*, vol. 7, pp. 1–11. American Mathematical Society, Providence, RI (1963)
6. El Ghami, M., Bai, Y.Q., Roos, C.: Kernel-functions based algorithms for semidefinite optimization. *RAIRO-Oper. Res.* **43**, 189–199 (2009)
7. El Ghami, M., Roos, C., Steihaug, T.: A generic primal-dual interior-point method for semidefinite optimization based on a new class of kernel functions. *Optimization Methods and Software* **25**(3), 387–403 (2010)
8. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. of the ACM* **42**(6), 1115–1145 (1995)
9. Helmberg, C., Rendl, F., Vanderbei, R.J., Wolkowicz, H.: An interior-point method for semidefinite programming. *SIAM J. on Optimization* **6**, 342–361 (1996)
10. Kojima, M., Shindoh, S., Hara, S.: Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices. *SIAM J. on Optimization* **7**, 86–125 (1997)
11. Liu, H., Liu, S., Xu, F.: A tight semidefinite relaxation of the max-cut problem. *J. of Combinatorial Optimization* **7**, 237–245 (2004)
12. Nesterov, Y.E., Nemirovskii, A.S.: *Interior Point Polynomial Algorithms in Convex Programming: Theory and Applications*. SIAM, Philadelphia, PA (1994)
13. Nesterov, Y.E., Todd, M.J.: Self-scaled barriers and interior-point methods for convex programming. *Mathematics of Operations Research* **22**, 1–42 (1997)
14. Nesterov, Y.E., Todd, M.J.: Primal-dual interior-point methods for self-scaled cones. *SIAM J. on Optimization* **8**, 324–364 (1998)
15. Peng, J., Roos, C., Terlaky, T.: Self-regular proximities and new search directions for linear and semidefinite optimization. *Mathematical Programming* **93**, 129–171 (2002)
16. Peng, J., Roos, C., Terlaky, T.: A new class of polynomial primal-dual methods for linear and semidefinite optimization. *European J. Operational Research* **143**(2), 234–256 (2002)
17. Peng, J., Roos, C., Terlaky, T.: *Self-Regularity: A New Paradigm of Primal-Dual Interior Point Algorithms*. Princeton University Press, Princeton (2002)
18. Peyghami, M.R.: An interior point approach for semidefinite optimization using new proximity functions. *Asia-Pacific J. of Operational Research* **26**(3), 365–382 (2009)
19. Roos, C., Terlaky, T., Vial, J.P.: *Interior Point Algorithms for Linear Optimization*. Second ed., Springer Science (2005)
20. Terlaky, T., Li, Y.: A new class of large neighborhood path-following interior point algorithms for semidefinite optimization with  $O\left(\sqrt{n} \log \frac{\text{Tr}(X^0 S^0)}{\epsilon}\right)$  iteration complexity. *SIAM J. on Optimization* **20**(6), 2853–2875 (2010)
21. Vandenberghe, L., Boyd, S.: Semidefinite programming. *SIAM Review* **38**, 49–95 (1996)
22. Wang, G.Q., Bai, Y.Q., Roos, C.: Primal-dual interior-point algorithms for semidefinite optimization based on simple kernel functions. *J. of Mathematical Modelling and Algorithms* **4**, 409–433 (2005)
23. Wolkowicz, H., Saigal, R., Vandenberghe, L.: *Handbook of Semidefinite Programming: Theory, Algorithms and Applications*. Kluwer Academic Publishers (2000)
24. Zhang Y.: On extending some primal-dual interior-point algorithms from linear programming to semidefinite programming. *SIAM J. on Optimization* **8**, 365–386 (1998)

# Chapter 16

## Elementary Optimality Conditions for Nonlinear SDPs

Florian Jarre

### 16.1 Introduction

Recent applications in areas such as free material optimization [6], robust control [4], passive reduced order modeling [5], combinatorial optimization [9], and others rely on the solution of nonlinear semidefinite programs. An implementation for solving such problems is presented in [7], for example.

This chapter generalizes optimality conditions for nonlinear programs to nonlinear semidefinite programs, highlighting some parallels and some differences. In Sect. 16.2.1 the problems and notation are introduced. Then, constraint qualifications for both programs are introduced. First order optimality conditions are presented for the case where a constraint qualification is satisfied. Finally, second order conditions are presented, where in addition, strict complementarity is assumed.

### 16.2 First Order Conditions

#### 16.2.1 NLSDPs and NLPs

This section concerns nonlinear semidefinite optimization problems of the form

$$\begin{aligned} & \text{minimize } f(x) \mid F(x) = 0, \\ & G(x) \preceq 0. \end{aligned} \tag{16.1}$$

---

F. Jarre (✉)

Department of Mathematics, University of Düsseldorf, Germany  
e-mail: [jarre@opt.uni-duesseldorf.de](mailto:jarre@opt.uni-duesseldorf.de)

Throughout, we assume that  $f : \mathbf{R}^n \rightarrow \mathbf{R}$ ,  $F : \mathbf{R}^n \rightarrow \mathbf{R}^m$ , and  $G : \mathbf{R}^n \rightarrow \mathcal{S}^d$  are continuously differentiable. The inequality  $G(x) \leq 0$  means that the matrix  $G(x)$  is required to be negative semidefinite.  $\mathcal{S}^d$  stands for the space of real symmetric  $d \times d$ -matrices, and  $\mathcal{S}_+^d$  denotes the cone of positive semidefinite matrices in  $\mathcal{S}^d$ . The scalar product of two matrices is denoted by  $A \bullet B = \text{trace}(A^T B) = \sum_{i,j} A_{i,j} B_{i,j}$ .

For comparison we also consider nonlinear optimization problems of the form

$$\begin{aligned} & \text{minimize } f(x) \mid F(x) = 0, \\ & G(x) \leq 0, \end{aligned} \tag{16.2}$$

with componentwise<sup>1</sup> inequalities  $G(x) \leq 0$ .

This section will emphasize the differences between NLSDPs of the form (16.1) and NLPs of the form (16.2). In particular, we emphasize two major points:

- First, the relation of tangential and linearized cones does not translate in a straightforward fashion from (16.2) to (16.1).
- And second, at an optimal solution, the curvature in the Lagrangian “differs substantially” in (16.2) and in (16.1).

The derivatives of  $f$  and  $F$  at a point  $x$  are, as usual, represented by a row vector  $Df(x)$  and the  $m \times n$  Jacobian matrix  $DF(x)$ . The entries of the vector  $F(x)$  will be denoted by  $F_\nu(x)$  for  $1 \leq \nu \leq m$  and the first and second derivatives of  $F_\nu$  by  $DF_\nu(x)$  and  $D^2F_\nu(x)$ . Thus,  $DF_\nu(x)$  is the  $\nu$ -th row of  $DF(x)$ .

The derivative of  $G$  at a point  $x$  is a linear map  $DG(x) : \mathbf{R}^n \rightarrow \mathcal{S}^d$ . When applying the linear map  $DG(x)$  to a vector  $\Delta x$  we use the notation  $DG(x)[\Delta x] \in \mathcal{S}^d$ , or

$$DG(x)[\Delta x] = \sum_{i=1}^n \Delta x_i G_i(x) \quad \text{where} \quad G_i(x) := \frac{\partial}{\partial x_i} G(x) \in \mathcal{S}^d.$$

$DG(x)$  satisfies the characteristic equation  $G(x + \Delta x) \approx G(x) + DG(x)[\Delta x]$ . The matrix entries of  $G_i(x)$  will be referred to by  $(G_i(x))_{k,l}$ . As a short hand notation we will write

$$(DG(x)[\Delta x])_{k,l} = \sum_{i=1}^n (G_i(x))_{k,l} \Delta x_i =: (DG(x))_{k,l} \Delta x$$

where  $(DG(x))_{k,l}$  is a row vector with entries  $(G_i(x))_{k,l}$ .

---

<sup>1</sup>Assuming that  $G(x)$  is a symmetric matrix, and that the “off-diagonal inequalities” are thus listed twice among the inequalities  $G(x) \leq 0$ , introduces some complication in the discussion of nondegeneracy for (16.2). Below, we will work with Lagrange multipliers being symmetric as well, thus “counting” the off-diagonal inequalities just once. To make things short, we may simply ignore the fact that formally, (16.2) contains redundant constraints.

### 16.2.2 Linearized Subproblems

Assume that some point  $\bar{x} \in \mathbf{R}^n$  is given. Linearizing  $f, F$ , and  $G$  in (16.1) about  $\bar{x}$  yields the *linearized semidefinite programming problem*:

$$\begin{aligned} \text{minimize } & f(\bar{x}) + Df(\bar{x})\Delta x \mid F(\bar{x}) + DF(\bar{x})\Delta x = 0, \\ & G(\bar{x}) + DG(\bar{x})[\Delta x] \leq 0. \end{aligned} \quad (16.3)$$

Here, we maintain the conic constraint “ $\leq 0$ ” and do not use a linearization involving the tangential cone to the set of negative semidefinite matrices. This allows a direct analysis of SQP type methods and of optimality conditions as discussed below.

Likewise, for (16.2), we obtain the linearized problem

$$\begin{aligned} \text{minimize } & f(\bar{x}) + Df(\bar{x})\Delta x \mid F(\bar{x}) + DF(\bar{x})\Delta x = 0, \\ & G(\bar{x}) + DG(\bar{x})[\Delta x] \leq 0. \end{aligned} \quad (16.4)$$

We recall that the first order optimality conditions of (16.1) or of (16.2) simply refer to the problems (16.3) or (16.4). It is therefore crucial to understand the relation of (16.1) and (16.3).

### 16.2.3 Tangential and Linearized Cones

We denote the feasible set of the NLSDP (16.1) by  $\mathcal{F}_1$  and recall that the tangential cone of  $\mathcal{F}_1$  at a point  $\bar{x} \in \mathcal{F}_1$  is given by

$$\mathcal{T}_1 := \{\Delta x \mid \exists s^k \rightarrow \Delta x, \exists \alpha_k > 0, \alpha_k \rightarrow 0 \text{ such that } \bar{x} + \alpha_k s^k \in \mathcal{F}_1\}. \quad (16.5)$$

It is straightforward to verify that  $Df(\bar{x})\Delta x \geq 0$  must hold true for all  $\Delta x \in \mathcal{T}_1$  if  $\bar{x}$  is a local minimizer of (16.1).

We denote the tangential cone of (16.3) at  $\Delta x = 0$  by  $\mathcal{L}_1$ .  $\mathcal{L}_1$  is the linearized cone of the NLSDP (16.1) at  $\bar{x}$ . As before,  $Df(\bar{x})\Delta x \geq 0$  must hold true for all  $\Delta x \in \mathcal{L}_1$  if  $\bar{x}$  is a minimizer of (16.3).

Thus, the relation of the first order optimality conditions of (16.1) and (16.3) depends on the relation of  $\mathcal{T}_1$  and  $\mathcal{L}_1$ .

For illustration we return to the NLP (16.2) and denote the feasible set of (16.2) by  $\mathcal{F}_2$  and the tangential cone at a point  $\bar{x} \in \mathcal{F}_2$  by  $\mathcal{T}_2$ . Likewise, let  $\mathcal{L}_2$  be the linearized cone of the NLP (16.2) at a point  $\bar{x}$ . The linearized cone is thus given by

$$\begin{aligned} \mathcal{L}_2 = \{ & \Delta x \mid F(\bar{x}) + DF(\bar{x})\Delta x = 0, \\ & G_{k,l}(\bar{x}) + (DG(\bar{x}))_{k,l}\Delta x \leq 0 \text{ for all } k, l \text{ with } G_{k,l}(\bar{x}) = 0 \}. \end{aligned} \quad (16.6)$$

A key argument in nonlinear optimization is based on the fact that  $\mathcal{T}_2 \subset \mathcal{L}_2$  always holds true. This fact can be established by a simple argument: For  $\Delta x \in \mathcal{T}_2$  we have by definition (16.5)

$$0 = F(\bar{x} + \alpha_k s^k) = F(\bar{x}) + \alpha_k DF(\bar{x})s^k + o(\alpha_k).$$

Dividing this by  $\alpha_k > 0$  and using the fact that  $F(\bar{x}) = 0$  yields

$$0 = DF(\bar{x})s^k + o(1),$$

and taking the limit as  $k \rightarrow \infty$  yields  $0 = DF(\bar{x})\Delta x$ , i.e.  $\Delta x$  satisfies the equality constraints in the definition of  $\mathcal{L}_2$ . Applying the same argument to the active inequalities  $G_{k,l}(x) \leq 0$  for all  $k, l$  with  $G_{k,l}(\bar{x}) = 0$ , we obtain that

$$\mathcal{T}_2 \subset \mathcal{L}_2 \tag{16.7}$$

for NLPs of the form (16.2).

### 16.2.3.1 Constraint Qualifications for NLP

- The converse direction  $\mathcal{L}_2 \subset \mathcal{T}_2$  can be shown, for example, when the MFCQ-constraint qualification for (16.2),

$$DF(\bar{x}) \text{ has linearly independent rows} \tag{16.8}$$

$$\exists \Delta x \text{ such that } DF(\bar{x})\Delta x = 0 \text{ and}$$

$$(DG(\bar{x}))_{k,l}\Delta x < 0 \quad \text{for all } (k, l) \text{ with } (G(\bar{x}))_{k,l} = 0, \tag{16.9}$$

is satisfied (see [8, 12, 13] or [11] Definition 12.5). In this case, since the above characterization of  $\mathcal{L}_2$  is polyhedral, also the set

$$\mathcal{T}_2 \text{ is polyhedral.}$$

- For later reference we also recall the linear independence constraint qualification LICQ for problem (16.2),

$$DF_\nu(\bar{x}) \text{ and } (DG(\bar{x}))_{k,l} \text{ are linearly independent} \tag{16.10}$$

for  $1 \leq \nu \leq m$  and  $(k, l)$  with  $k \leq l$  and  $(G(\bar{x}))_{k,l} = 0$ . It is easy to see that (16.10) implies (16.8), (16.9).

Note that the semidefinite constraint  $G(x) \leq 0$  can be represented by an exponential (but finite!) number of smooth nonlinear inequality constraints, namely that the determinants of all principal submatrices of “ $-G(x)$ ” be nonnegative. Thus, one

might expect, that the tangential cone  $\mathcal{T}_1$  is also a polyhedral cone. As we will see, this is not the case! (The determinant will not satisfy MFCQ (16.9) when zero is an eigenvalue of multiplicity more than one.)

We derive a representation of  $\mathcal{T}_1$ : Let  $\bar{x} \in \mathcal{F}_1$  be given and, as in Definition (16.5), a sequence  $\alpha_k > 0$  with  $\alpha_k \rightarrow 0$ , and a sequence  $s^k \rightarrow \Delta x$ , such that  $F(\bar{x} + \alpha_k s^k) = 0$  and  $G(\bar{x} + \alpha_k s^k) \leq 0$  for all  $k$ .

As before it follows that  $DF(\bar{x})\Delta x = 0$ . Let

$$G(\bar{x}) = U \Lambda U^T = \begin{pmatrix} U^{(1)} & U^{(2)} \end{pmatrix} \begin{pmatrix} \Lambda^{(1)} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} (U^{(1)})^T \\ (U^{(2)})^T \end{pmatrix}$$

be the eigenvalue decomposition of  $G(\bar{x})$  where  $\Lambda^{(1)}$  is negative definite. Using the Schur complement it is straightforward to verify that the tangential cone of “ $-\mathcal{S}_+^d$ ” at  $G(\bar{x})$  is given by all matrices of the form

$$W = \begin{pmatrix} U^{(1)} & U^{(2)} \end{pmatrix} \begin{pmatrix} * & * \\ * & \tilde{W}^{(2)} \end{pmatrix} \begin{pmatrix} (U^{(1)})^T \\ (U^{(2)})^T \end{pmatrix} \quad (16.11)$$

where  $\tilde{W}^{(2)} \leq 0$  and “\*” can be anything. We keep  $U$  fixed and define  $\tilde{G}(x) := U^T G(x) U$ . (Then,  $G(x) \leq 0$  iff  $\tilde{G}(x) \leq 0$ .) We partition

$$\tilde{G}(x) := \begin{pmatrix} \tilde{G}^{(1)}(x) & \tilde{G}^{(1,2)}(x) \\ \tilde{G}^{(1,2)}(x)^T & \tilde{G}^{(2)}(x) \end{pmatrix}. \quad (16.12)$$

conforming the partition in (16.11). (Thus,  $\tilde{G}^{(1)}(\bar{x})$  and  $\Lambda^{(1)}$  coincide and  $\tilde{G}^{(1,2)}(\bar{x}) = 0$ ,  $\tilde{G}^{(2)}(\bar{x}) = 0$ .) Since  $\tilde{G}(\bar{x} + \alpha_k s^k) \leq 0$  it follows that

$$0 \geq \tilde{G}^{(2)}(\bar{x} + \alpha_k s^k) - \underbrace{\tilde{G}^{(2)}(\bar{x})}_{=0} \approx \alpha_k D\tilde{G}^{(2)}(\bar{x})[s^k].$$

Dividing by  $\alpha_k > 0$  and taking the limit as  $k \rightarrow \infty$ , we get

$$D\tilde{G}^{(2)}(\bar{x})[\Delta x] \leq 0,$$

i.e., by (16.11),  $G(\bar{x})[\Delta x]$  lies in the tangential cone of “ $-\mathcal{S}_+^d$ ” at  $G(\bar{x})$ . The above implies that

$$\mathcal{T}_1 \subset \{\Delta x \mid DF(\bar{x})\Delta x = 0, \quad D\tilde{G}^{(2)}(\bar{x})[\Delta x] \leq 0\} \quad (16.13)$$

in analogy to (16.7). By linearizing the equality constraints and the “active part” of the inequalities, the set on the right hand side appears to be a straightforward generalization of the definition of  $\mathcal{L}_2$  in (16.6). It turns out, however, that the set on the right hand side is not necessarily  $\mathcal{L}_1$ :

The example  $x \in \mathbf{R}$ ,

$$G(x) := \begin{pmatrix} -2 & x \\ x & -x^2 \end{pmatrix} \leq 0$$

shows that  $\mathcal{F}_1 = \mathbf{R}$ , and thus, also  $\mathcal{T}_1 = \mathbf{R}$ , while the feasible set of (16.3) at  $\bar{x} = 0$  is given by

$$\left\{ \Delta x \left| \begin{pmatrix} -2 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & \Delta x \\ \Delta x & 0 \end{pmatrix} \leq 0 \right. \right\} = \{0\}. \quad (16.14)$$

Hence, also the linearized cone is given by  $\mathcal{L}_1 = \{0\}$  and does not contain  $\mathcal{T}_1$  in contrast to (16.7).

#### 16.2.4 A Constraint Qualification for NLSDP

We return to the NLSDP (16.1). The MFCQ condition (16.8), (16.9) allows a straightforward extension: Problem (16.1) satisfies Robinson's constraint qualification [12] (or a generalized MFCQ) at a point  $\bar{x} \in \mathcal{F}$ , if

$DF(\bar{x})$  has linearly independent rows, and there exists a vector  $d$  such that

$$DF(\bar{x})d = 0 \text{ and}$$

$$G(\bar{x}) + DG(\bar{x})[d] < 0. \quad (16.15)$$

The last two lines of (16.15) coincide with Slater's condition for (16.3).

The modified example  $x \in \mathbf{R}^2$ , and

$$G(x) := \begin{pmatrix} -2 & x_1 \\ x_1 & x_2 - x_1^2 \end{pmatrix} \leq 0$$

coincides with the above example (leading to (16.14)) when  $x_2$  is set to 0. However, this example satisfies (16.15) at  $\bar{x} = (0, 0)^T$ , and the feasible set of (16.3) at  $\bar{x} = 0$  is given by

$$\left\{ \Delta x \left| \begin{pmatrix} -2 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & \Delta x_1 \\ \Delta x_1 & \Delta x_2 \end{pmatrix} \leq 0 \right. \right\} = \{\Delta x \mid \Delta x_1^2 \leq -2\Delta x_2\}.$$

Straightforward calculations show that the tangential cone of this set is given by  $\mathcal{L}_1 = \{\Delta x \mid \Delta x_2 \leq 0\}$  and that  $\mathcal{T}_1 = \mathcal{L}_1$ . This is true more generally, as is made explicit in the next lemma.

**Lemma 16.1.** *If the constraint qualification (16.15) is satisfied, then  $\mathcal{L}_1 = \mathcal{T}_1$  coincides with the right hand side of (16.13).*

Lemma 16.1 establishes the earlier remark that the tangential cone of an NLSPD is not polyhedral, in general, in contrast to the tangential cone of an NLP. This concludes our discussions of the first “major point” referred to in Sect. 16.2.1.

Lemma 16.1 and the above discussions are known for a long time, see the results of Robinson, [12], but they may not be widely known. As conic optimization has returned to the focus of research following the generalization of interior-point methods to semidefinite programming, [1, 10, 15] these conditions now become of wider general interest.

### 16.2.5 KKT Conditions

For the remainder of this chapter we assume that the constraint qualification (16.15) at  $\bar{x}$  is always satisfied. We assume that  $\bar{x}$  is a local minimizer of (16.1). Lemma 16.1 implies that the KKT conditions of (16.3) for  $\Delta x = 0$  can be translated to (16.1). This yields the following first order conditions:

**Theorem 16.1.** *Let  $\bar{x}$  be a local minimizer of (16.1) and let (16.1) be regular at  $\bar{x}$  in the sense of (16.15). Then there exists a matrix  $\bar{Y} \in \mathcal{S}_+^d$  and a vector  $\bar{y} \in \mathbf{R}^m$  such that*

$$Df(\bar{x})^T + DF(\bar{x})^T \bar{y} + \begin{pmatrix} G_1(\bar{x}) \bullet \bar{Y} \\ \vdots \\ G_m(\bar{x}) \bullet \bar{Y} \end{pmatrix} = 0 \quad \text{and} \quad G(\bar{x}) \bullet \bar{Y} = 0. \quad (16.16)$$

The Lagrangian function underlying Theorem 16.1 is given by

$$L(x, y, Y) := f(x) + F(x)^T y + G(x) \bullet Y \quad (16.17)$$

with  $Y \in \mathcal{S}_+^d$ , and the first equation in (16.16) states that the derivative  $D_x L(\bar{x}, \bar{y}, \bar{Y})$  of the Lagrangian with respect to  $x$  is zero at the optimal solution,

$$D_x L(\bar{x}, \bar{y}, \bar{Y}) = 0.$$

The Lagrangian for the NLP (16.2) is identical to (16.17) except that we have  $Y \in \mathcal{S}^d$ ,  $Y \geq 0$  for (16.2). The fact that the Lagrangian of (16.1) and of (16.2) is the same function (just with a different domain for  $Y$ ) leads us to the second crucial difference of the NLSDP (16.1) and the NLP (16.2).

## 16.3 Second Order Conditions

In this section we assume that  $f, F$ , and  $G$  are all twice continuously differentiable. Here, the second derivative  $D^2G$  at a point  $x \in \mathbf{R}^n$  is a symmetric bilinear mapping,

$$D^2G(x) : \mathbf{R}^n \times \mathbf{R}^n \rightarrow \mathcal{S}^d, \quad D^2G(x)[\Delta x, \Delta x] = \sum_{i,j=1}^n \Delta x_i \Delta x_j G_{i,j}(x) \in \mathcal{S}^d$$

with  $G_{i,j}(x) := \frac{\partial^2}{\partial x_i \partial x_j} G(x)$ , and for  $Y \in \mathcal{S}^d$  we have

$$Y \bullet D^2G(x)[\Delta x, \Delta x] = \sum_{i,j=1}^n \Delta x_i \Delta x_j (Y \bullet G_{i,j}(x)) = \Delta x^T H \Delta x,$$

where  $H = H(x, Y)$  is the symmetric  $n \times n$ -matrix with matrix entries  $Y \bullet G_{i,j}(x)$ .

### 16.3.1 Cone of Critical Directions

Let  $\bar{x}$  be a local minimizer of the NLSDP (16.1) and let the generalized MFCQ condition (16.15) be satisfied at  $\bar{x}$ .

Then, the KKT conditions state that  $Df(\bar{x})\Delta x \geq 0$  for all  $\Delta x \in \mathcal{T}_1$ . Clearly, if  $Df(\bar{x})\Delta x > 0$ , then  $\Delta x$  is an ascent direction for  $f$ . The cone of critical directions is given by

$$C_1 := \{\Delta x \in \mathcal{T}_1 \mid Df(\bar{x})\Delta x = 0\}.$$

Let  $\Delta x \in C_1$  ( $\Delta x \neq 0$ ) be given. By the generalized MFCQ (16.15) there exist differentiable curves  $x(t) \in \mathcal{F}_1$  with  $x(0) = \bar{x}$  and  $\dot{x}(0) = \Delta x$ . Since  $Df(\bar{x})\dot{x}(0) = 0$  the question whether  $f$  is locally increasing or decreasing along such a curve depends on the second order terms in the constraints and in the objective function.

For the case of an NLP of the form (16.2) the cone of critical directions (short: critical cone) depends on three conditions:

$$DF(\bar{x})\Delta x = 0, \tag{16.18}$$

$$(DG(\bar{x}))_{k,l}\Delta x = 0 \quad \text{for all } (k, l) \text{ with } \bar{Y}_{k,l} > 0 \tag{16.19}$$

$$(DG(\bar{x}))_{k,l}\Delta x \leq 0 \quad \text{for all } (k, l) \text{ with } (G(\bar{x}))_{k,l} = 0, \bar{Y}_{k,l} = 0. \tag{16.20}$$

Note that the condition  $Df(\bar{x})\Delta x = 0$  does not need to be included above since, by the KKT conditions,  $Df(\bar{x})$  is spanned by the vectors in (16.18) and (16.19). The set of vectors  $\Delta x$  with (16.18)–(16.20) always contains the critical cone; and if LICQ (16.10) is satisfied, it coincides with the critical cone. If we assume LICQ and strict complementarity, then condition (16.20) will not apply, and the critical cone is in fact a linear subspace,

$$C_2 = \{\Delta x \mid (16.18) \text{ and } (16.19) \text{ hold}\}.$$

Returning to the NLSDP (16.1) we also assume uniqueness of the multipliers, strict complementarity,  $G(\bar{x}) - \bar{Y} < 0$ , and a constraint qualification that is detailed in the next section. Proceeding in an analogous fashion, namely by replacing the inequalities  $D\tilde{G}^{(2)}(\bar{x})[\Delta x] \leq 0$  associated with the positive part of  $\bar{Y}$  with equalities, the critical cone is given by

$$C_1 = \{\Delta x \mid DF(\bar{x})\Delta x = 0, \quad (U^{(2)})^T DG(\bar{x})[\Delta x]U^{(2)} = 0\}. \quad (16.21)$$

Let  $U^{(2)}$  have  $q$  columns, i.e.  $U^{(2)}$  is a  $d \times q$  matrix. It turns out that  $C_1$  is the linearized cone of the following boundary manifold of the feasible set

$$\mathcal{F}_1^{bd} := \{x \mid F(x) = 0, \quad \text{rank}(G(x)) = d - q\} \quad (16.22)$$

at  $\bar{x}$ . Locally, near  $\bar{x}$ , this is a subset of  $\mathcal{F}_1$  (since the negative eigenvalues of  $G(x)$  remain negative for small  $\|x - \bar{x}\|$ ). In order to guarantee coincidence of the tangential cone of  $\mathcal{F}_1^{bd}$  with  $C_1$  we require that  $\mathcal{F}_1^{bd}$  satisfies some constraint qualification at  $\bar{x}$ . Yet, on first sight, it might not be quite obvious how to define a constraint qualification for the rank condition.

### 16.3.2 A Second Order Constraint Qualification

Recall the definition  $\tilde{G}(x) := U^T G(x)U$  and its partition (16.12) and note that  $\tilde{G}^{(1)}(x) < 0$  for small  $\|x - \bar{x}\|$ .

The condition that  $G(x)$  (and thus also  $\tilde{G}(x)$ ) has rank  $d - q$  (see (16.22)) therefore translates to the condition that the Schur complement

$$\tilde{G}^{(2)}(x) - \tilde{G}^{(1,2)}(x)^T (\tilde{G}^{(1)}(x))^{-1} \tilde{G}^{(1,2)}(x) \equiv 0 \quad (16.23)$$

equals zero (for small  $\|x - \bar{x}\|$ ).

Keeping  $U$  and the partition of  $\tilde{G}$  fixed, the above representation of the constraint “ $\text{rank}(G(x)) = d - q$ ” is simply a system of  $q(q + 1)/2$  nonlinear equations. Hence, MFCQ (16.8), (16.9) coincides with LICQ (16.10), and the tangential cone of  $\mathcal{F}_1^{bd}$  at  $\bar{x}$  always is a subset of  $C_1$ . Since  $\tilde{G}^{(1,2)}(\bar{x}) = 0$ , the derivative of (16.23) at  $x = \bar{x}$  is given by

$$D\tilde{G}^{(2)}(\bar{x}) = D_x \left( (U^{(2)})^T G(x) U^{(2)} \right) \Big|_{x=\bar{x}}. \quad (16.24)$$

Regularity of  $\mathcal{F}_1^{bd}$  follows, for example, if the  $q(q + 1)/2$  gradients of the constraints in (16.24) and the  $m$  gradients of  $F_\nu(x)$  at  $\bar{x}$ , ( $1 \leq \nu \leq m$ ) are linearly independent.

This requirement, however, may be too strong; in particular, in the case that  $G$  is diagonal or block diagonal. In this case, also the matrix  $U$  can be chosen as a block diagonal matrix, and it suffices to check linear independence of the gradients associated with  $F_\nu$  ( $1 \leq \nu \leq m$ ) and with the  $U^{(2)}$ -parts of the diagonal blocks of  $G$ .

Likewise, for example, when  $G$  is a congruence transformation of a block diagonal matrix, the requirement of linear independence can also be restricted to a smaller subset of active gradients, but if the transformation is not known explicitly, this subset may be difficult to identify. In order not to exclude such cases we generalize the requirement above and simply define that problem (16.1) satisfies the second order constraint qualification if the linearized cone of  $\mathcal{F}_1^{bd}$  coincides with the right hand side of (16.21).

### 16.3.3 A Second Example

For problem (16.2), if  $\bar{x}$  is a local minimizer satisfying LICQ (16.10), then the Hessian of the Lagrangian is positive semidefinite on  $C_2$ , and conversely, if a KKT point  $\bar{x}$  of (16.2) is given such that the Hessian of the Lagrangian is positive definite on  $C_2$ , then  $\bar{x}$  is a strict local minimizer of (16.2).

Again, this property does not translate to (16.1). As an example by Diehl et.al. [3] illustrates, the Hessian of the Lagrangian may be negative definite even if the cone of critical directions contains nonzero elements: Consider the problem of two variables

$$\text{minimize } -x_1^2 - (1-x_2)^2 \mid \begin{pmatrix} -1 & x_1 & x_2 \\ x_1 & -1 & 0 \\ x_2 & 0 & -1 \end{pmatrix} \leq 0. \quad (16.25)$$

The semidefiniteness constraint is satisfied if, and only if,  $x_1^2 + x_2^2 \leq 1$ . Thus, the global optimal solution is easily identified as  $\bar{x} = (0, -1)^T$ .

If we consider the equivalent NLP problem with the  $1 \times 1$ -matrix-constraint  $G(x) := x_1^2 + x_2^2 - 1 \leq 0$ , this minimizer is “perfectly fine”. (It satisfies strict complementarity, LICQ (16.10), and second order growth condition.)

Since the semidefiniteness constraint in (16.25) is linear, the Hessian of the Lagrangian of (16.25) is given by the Hessian of  $f$  and is thus negative definite everywhere, also the cone of critical directions which is given by all vectors of the form  $(x_1, 0)^T$  with  $x_1 \in \mathbf{R}$ .

As pointed out in [3], this lack of semidefiniteness has implications: A sequential semidefinite programming algorithm that uses subproblems with a *convex* quadratic objective function and linearized semidefiniteness constraints generally will not converge superlinearly, no matter how the convex objective function is chosen. (And subproblems with a *nonconvex* quadratic objective function and linear semidefiniteness constraints are very hard to solve.)

This is in strong contrast to sequential quadratic programming algorithms for NLPs of the form (16.2) that do converge quadratically under standard assumptions – also when suitable convex quadratic objective functions are being used for the subproblems.

### 16.3.4 Second Order Necessary and Sufficient Conditions

The main result of this section is stated in the following theorems:

**Theorem 16.2.** *Let  $\bar{x}$  be a local minimizer of (16.1) and let  $\bar{x}, \bar{y}, \bar{Y}$  be a strictly complementary KKT-point. Assume that problem (16.1) satisfies the second order constraint qualification of Sect. 16.3.2. Then*

$$h^T(D_x^2L(\bar{x}, \bar{y}, \bar{Y}) + K(\bar{x}, \bar{Y}))h \geq 0 \quad \forall h \in C_1,$$

where  $K(\bar{x}, \bar{Y}) \succeq 0$  is a matrix depending on the curvature of the semidefinite cone at  $G(\bar{x})$  and the directional derivatives of  $G$  at  $\bar{x}$ , and is given by its matrix entries

$$(K(\bar{x}, \bar{Y}))_{i,j} := -2\bar{Y} \bullet G_i(\bar{x})G_j(\bar{x})^\dagger G_j(\bar{x})^T.$$

The converse direction also holds without assuming regularity of  $\mathcal{F}_1^{bd}$ :

**Theorem 16.3.** *Let  $\bar{x}$  be a strictly complementary KKT-point of (16.1) and assume that*

$$h^T(D_x^2L(\bar{x}, \bar{y}, \bar{Y}) + K(\bar{x}, \bar{Y}))h > 0 \quad \forall h \in C_1.$$

*Then,  $\bar{x}$  is a strict local minimizer of (16.1) that satisfies the second order growth condition,*

$$\exists \epsilon > 0, \delta > 0 : f(x + s) \geq f(x) + \epsilon \|s\|^2 \quad \forall \|s\| \leq \delta \text{ with } x + s \in \mathcal{F}_1.$$

A stronger second order sufficient condition requiring that  $D_x^2L(\bar{x}, \bar{y}, \bar{Y})$  be positive definite on  $C_1$  was already given in [13, Definition 2.1], while the weaker form above is due to [14].

The weak form complementing the necessary condition includes the “extra term”  $h^T K(\bar{x}, \bar{Y})h$  which explains the observation in (16.25) that the Hessian of the Lagrangian may be negative definite at the optimal solution, even if  $C_1$  contains nonzero elements. For completeness we present a self-contained proof of both results:

*Proof (of Theorem 16.2).* Assume that  $\bar{x}$  is a local minimizer of (16.1) and that (16.1) satisfies the second order constraint qualification of Sect. 16.3.2. Let  $\Delta x \in C_1$  be given. Then there exist  $\alpha_k > 0$ ,  $\alpha_k \rightarrow 0$ ,  $s^k \rightarrow \Delta x$  with  $\bar{x} + \alpha_k s^k \in \mathcal{F}_1^{bd}$ . By assumption,

$$0 \leq f(\bar{x} + \alpha_k s^k) - f(\bar{x})$$

for all sufficiently large  $k$ . Observe that the right hand side above equals

$$\alpha_k Df(\bar{x})s^k + \frac{\alpha_k^2}{2}(s^k)^T D^2f(\bar{x})s^k + o(\alpha_k^2).$$

Inserting the KKT conditions, namely,

$$Df(\bar{x})s^k = -\bar{y}^T DF(\bar{x})s^k - \bar{Y} \bullet DG(\bar{x})[s^k]$$

into this expression, and canceling a factor  $\alpha_k$ , we obtain

$$0 \leq -\bar{y}^T DF(\bar{x})s^k - \bar{Y} \bullet DG(\bar{x})[s^k] + \frac{\alpha_k}{2}(s^k)^T D^2 f(\bar{x})s^k + o(\alpha_k). \quad (16.26)$$

From

$$0 = F_\nu(\bar{x} + \alpha_k s^k) = \underbrace{F_\nu(\bar{x})}_{=0} + \alpha_k DF_\nu(\bar{x})s^k + \frac{\alpha_k^2}{2}(s^k)^T D^2 F_\nu(\bar{x})s^k + o(\alpha_k^2)$$

it follows

$$-DF_\nu(\bar{x})s^k = \frac{\alpha_k}{2}(s^k)^T D^2 F_\nu(\bar{x})s^k + o(\alpha_k).$$

Inserting this in (16.26) yields

$$0 \leq -\bar{Y} \bullet DG(\bar{x})[s^k] + \frac{\alpha_k}{2}(s^k)^T \left( D^2 f(\bar{x}) + \sum_{\nu=1}^m \bar{y}_\nu D^2 F_\nu(\bar{x}) \right) s^k + o(\alpha_k). \quad (16.27)$$

In analogy to the transformation  $\tilde{G}(x) = U^T G(x)U$ , we define  $\tilde{Y} = U^T \bar{Y} U$ . Note that by complementarity, only  $\tilde{Y}^{(2)}$  is nonzero. Using (16.23) (in line 5, below) it follows that

$$\begin{aligned} & \underbrace{\bar{Y} \bullet G(\bar{x}) + \alpha_k \bar{Y} \bullet DG(\bar{x})[s^k] + \frac{\alpha_k^2}{2} \bar{Y} \bullet D^2 G(\bar{x})[s^k, s^k]}_{=0} + o(\alpha_k^2) \\ &= \bar{Y} \bullet G(\bar{x} + \alpha_k s^k) \\ &= \tilde{Y} \bullet \tilde{G}(\bar{x} + \alpha_k s^k) \\ &= \tilde{Y}^{(2)} \bullet \tilde{G}^{(2)}(\bar{x} + \alpha_k s^k) \\ &= \tilde{Y}^{(2)} \bullet \left( \tilde{G}^{(1,2)}(\bar{x} + \alpha_k s^k)^T (\tilde{G}^{(1)}(\bar{x} + \alpha_k s^k))^{-1} \tilde{G}^{(1,2)}(\bar{x} + \alpha_k s^k) \right) \\ &= \alpha_k^2 \tilde{Y}^{(2)} \bullet \left( (D\tilde{G}^{(1,2)}(\bar{x})[s^k])^T (\tilde{G}^{(1)}(\bar{x}))^{-1} D\tilde{G}^{(1,2)}(\bar{x})[s^k] \right) + o(\alpha_k^2). \end{aligned} \quad (16.28)$$

In line 6 above (the last line) we used that fact that  $\tilde{G}^{(1,2)}(\bar{x}) = 0$ , so that

$$\tilde{G}^{(1,2)}(\bar{x} + \alpha_k s^k) = \alpha_k D\tilde{G}^{(1,2)}(\bar{x})[s^k] + o(\alpha_k^2).$$

Dividing the first and the last term in the above chain of equalities by  $\alpha_k$  we obtain.

$$\begin{aligned} & \bar{Y} \bullet DG(\bar{x})[s^k] + \frac{\alpha_k}{2} \bar{Y} \bullet D^2 G(\bar{x})[s^k, s^k] + o(\alpha_k) \\ &= \alpha_k \tilde{Y}^{(2)} \bullet \left( (D\tilde{G}^{(1,2)}(\bar{x})[s^k])^T (\tilde{G}^{(1)}(\bar{x}))^{-1} D\tilde{G}^{(1,2)}(\bar{x})[s^k] \right) + o(\alpha_k) \\ &= \alpha_k \tilde{Y} \bullet \left( (D\tilde{G}(\bar{x})[s^k])^T (\tilde{G}(\bar{x}))^\dagger D\tilde{G}(\bar{x})[s^k] \right) + o(\alpha_k), \end{aligned}$$

where  $G^\dagger$  denotes the pseudo inverse of a matrix  $G$ . In the last line above, we use the fact that  $\tilde{G}(\bar{x})$  is zero outside the  $\tilde{G}^{(1)}(\bar{x})$ -block and  $\tilde{Y}$  is zero outside the  $\tilde{Y}^{(2)}$ -block. Undoing the change of basis (i.e. multiplying all matrices  $\tilde{G}$  and  $\tilde{Y}$  by  $U$  and by  $U^T$  from left and right) yields

$$\begin{aligned} & -\bar{Y} \bullet DG(\bar{x})[s^k] \\ &= \frac{\alpha_k}{2} \bar{Y} \bullet \left( D^2 G(\bar{x})[s^k, s^k] - 2DG(\bar{x})[s^k]^T (G(\bar{x}))^\dagger DG(\bar{x})[s^k] \right) + o(\alpha_k). \end{aligned}$$

Inserting this in (16.27) finally yields

$$\begin{aligned} 0 &\leq \frac{\alpha_k}{2} (s^k)^T \left( D^2 f(\bar{x}) + \sum_{v=1}^m \bar{y}_v D^2 F_v(\bar{x}) \right) s^k \\ &+ \frac{\alpha_k}{2} \bar{Y} \bullet \left( D^2 G(\bar{x})[s^k, s^k] - 2DG(\bar{x})[s^k]^T (G(\bar{x}))^\dagger DG(\bar{x})[s^k] \right) + o(\alpha_k). \end{aligned}$$

Note that the right hand side can be written in terms of the Hessian of the Lagrangian (16.17),

$$0 \leq \frac{\alpha_k}{2} \left( (s^k)^T D^2 L(\bar{x}, \bar{y}, \bar{Y}) s^k - 2\bar{Y} \bullet DG(\bar{x})[s^k]^T (G(\bar{x}))^\dagger DG(\bar{x})[s^k] \right) + o(\alpha_k).$$

Dividing by  $\alpha_k$  and taking the limit as  $k \rightarrow \infty$ , we arrive at:

$$0 \leq \frac{1}{2} \left( (\Delta x)^T D^2 L(\bar{x}, \bar{y}, \bar{Y}) \Delta x - 2\bar{Y} \bullet DG(\bar{x})[\Delta x]^T (G(\bar{x}))^\dagger DG(\bar{x})[\Delta x] \right).$$

Summarizing,  $\bar{x}, \bar{y}, \bar{Y}$  satisfy the second order necessary condition of Theorem 16.2.  $\square$

*Proof (of Theorem 16.3).* Let the assumptions of Theorem 16.3 be satisfied and assume that  $\bar{x}$  is not a strict local minimizer as stated in the theorem, i.e. there exists a sequence of feasible points  $x^{(k)}$  with

$$\|x^{(k)} - \bar{x}\| \leq \frac{1}{k} \quad \text{and} \quad f(x^{(k)}) < f(\bar{x}) + \frac{1}{k} \|x^{(k)} - \bar{x}\|^2.$$

In particular,  $x^{(k)} \neq \bar{x}$  for all  $k$ . By considering a subsequence, one may assume without loss of generality that  $s^{(k)} := \frac{x^{(k)} - \bar{x}}{\|x^{(k)} - \bar{x}\|} \rightarrow \bar{s}$ . By construction,  $\bar{s}$  lies in the tangential cone  $\mathcal{T}_1$  of (16.1) at  $\bar{x}$  and therefore satisfies (16.13).

We show that  $\bar{s}$  also lies in the cone of critical directions (16.21) at  $\bar{x}$ . Let  $g(x) := \bar{Y} \bullet G(x)$ , so that for feasible points  $x$  we have  $L(x, \bar{y}, \bar{Y}) = f(x) + g(x)$ . Then,

$$\begin{aligned} \frac{1}{k} \|x^{(k)} - \bar{x}\|^2 &> f(x^{(k)}) - f(\bar{x}) \\ &= L(x^{(k)}, \bar{y}, \bar{Y}) - L(\bar{x}, \bar{y}, \bar{Y}) - (g(x^{(k)}) - g(\bar{x})) \end{aligned} \quad (16.29)$$

Using Taylors expansion and the fact that  $D_x L(\bar{x}, \bar{y}, \bar{Y}) = 0$  we obtain

$$\begin{aligned} L(x^{(k)}, \bar{y}, \bar{Y}) &= L(\bar{x}, \bar{y}, \bar{Y}) + \frac{1}{2}(x^{(k)} - \bar{x})^T \\ &\quad D_{xx}^2 L(\bar{x}, \bar{y}, \bar{Y})(x^{(k)} - \bar{x}) + o(\|x^{(k)} - \bar{x}\|^2). \end{aligned}$$

Dividing  $L(x^{(k)}, \bar{y}, \bar{Y}) - L(\bar{x}, \bar{y}, \bar{Y})$  by  $\|x^{(k)} - \bar{x}\|^2$  and using (16.29) we get

$$\frac{1}{k} > \frac{1}{2}(s^{(k)})^T D_{xx}^2 L(\bar{x}, \bar{y}, \bar{Y}) s^{(k)} - \frac{g(x^{(k)}) - g(\bar{x})}{\|x^{(k)} - \bar{x}\|^2} + o(1). \quad (16.30)$$

Since  $s^{(k)}$  converges and  $g(x^{(k)}) \leq 0 = g(\bar{x})$ , there exists  $M < \infty$  such that

$$0 \leq -\frac{g(x^{(k)}) - g(\bar{x})}{\|x^{(k)} - \bar{x}\|^2} \leq M.$$

Thus,

$$0 = \lim_{k \rightarrow \infty} \frac{g(x^{(k)}) - g(\bar{x})}{\|x^{(k)} - \bar{x}\|}. \quad (16.31)$$

Here,

$$\begin{aligned} g(x^{(k)}) - g(\bar{x}) &= Dg(\bar{x})(x^{(k)} - \bar{x}) + o(\|x^{(k)} - \bar{x}\|) \\ &= \bar{Y} \bullet DG(\bar{x})[x^{(k)} - \bar{x}] + o(\|x^{(k)} - \bar{x}\|). \end{aligned}$$

Inserting this in (16.31) and taking the limit yields

$$0 = Dg(\bar{x})\bar{s} = \bar{Y} \bullet DG(\bar{x})[\bar{s}].$$

Since  $\bar{s} \in \mathcal{T}_1$  it follows  $DF(\bar{x})\bar{s} = 0$  and from strict complementarity and the second relation in (16.13) it follows  $(U^{(2)})^T DG(\bar{x})[\bar{s}] U^{(2)} = 0$  and thus,  $\bar{s} \in C_1$ .

Starting from (16.29) we may now follow the chain of inequalities in the proof of Theorem (16.2) with reversed signs. With  $\alpha_k := \|x^{(k)} - \bar{x}\|$  we obtain as in (16.27)

$$\frac{\alpha_k}{k} \geq -\bar{Y} \bullet DG(\bar{x})[s^k] + \frac{\alpha_k}{2}(s^k)^T \left( D^2 f(\bar{x}) + \sum_{v=1}^m \bar{y}_v D^2 F_v(\bar{x}) \right) s^k + o(\alpha_k). \quad (16.32)$$

Since  $G(x^{(k)}) \leq 0$  we may continue as in (16.28),

$$\begin{aligned} & \underbrace{\bar{Y} \bullet G(\bar{x})}_{=0} + \alpha_k \bar{Y} \bullet DG(\bar{x})[s^k] + \frac{\alpha_k^2}{2} \bar{Y} \bullet D^2G(\bar{x})[s^k, s^k] + o(\alpha_k^2) \\ &= \tilde{Y}^{(2)} \bullet \tilde{G}^{(2)}(\bar{x} + \alpha_k s^k) \\ &\leq \tilde{Y}^{(2)} \bullet (\tilde{G}^{(1,2)}(\bar{x} + \alpha_k s^k)^T (\tilde{G}^{(1)}(\bar{x} + \alpha_k s^k))^{-1} \tilde{G}^{(1,2)}(\bar{x} + \alpha_k s^k)) \\ &= \alpha_k^2 \tilde{Y}^{(2)} \bullet ((D\tilde{G}^{(1,2)}(\bar{x})[s^k])^T (\tilde{G}^{(1)}(\bar{x}))^{-1} D\tilde{G}^{(1,2)}(\bar{x})[s^k]) + o(\alpha_k^2). \end{aligned}$$

Again, it follows

$$\begin{aligned} & \bar{Y} \bullet DG(\bar{x})[s^k] + \frac{\alpha_k}{2} \bar{Y} \bullet D^2G(\bar{x})[s^k, s^k] + o(\alpha_k) \\ &\leq \alpha_k \tilde{Y} \bullet ((D\tilde{G}(\bar{x})[s^k])^T (\tilde{G}(\bar{x}))^\dagger D\tilde{G}(\bar{x})[s^k]) + o(\alpha_k) \\ &= \alpha_k \bar{Y} \bullet ((DG(\bar{x})[s^k])^T (G(\bar{x}))^\dagger DG(\bar{x})[s^k]) + o(\alpha_k), \end{aligned}$$

leading to

$$\begin{aligned} \frac{\alpha_k}{k} &\geq \frac{\alpha_k}{2} (s^k)^T \left( D^2 f(\bar{x}) + \sum_{v=1}^m \bar{y}_v D^2 F_v(\bar{x}) \right) s^k \\ &\quad + \frac{\alpha_k}{2} \bar{Y} \bullet (D^2 G(\bar{x})[s^k, s^k] - 2DG(\bar{x})[s^k]^T (G(\bar{x}))^\dagger DG(\bar{x})[s^k]) + o(\alpha_k). \end{aligned}$$

where the right hand side can be written in terms of the Hessian of the Lagrangian,

$$\frac{\alpha_k}{k} \geq \frac{\alpha_k}{2} \left( (s^k)^T D^2 L(\bar{x}, \bar{y}, \bar{Y}) s^k - 2 \bar{Y} \bullet DG(\bar{x})[s^k]^T (G(\bar{x}))^\dagger DG(\bar{x})[s^k] \right) + o(\alpha_k).$$

Dividing by  $\alpha_k$  and taking the limit as  $k \rightarrow \infty$ , we arrive at:

$$0 \geq \frac{1}{2} \left( \bar{s}^T D^2 L(\bar{x}, \bar{y}, \bar{Y}) \bar{s} - 2 \bar{Y} \bullet DG(\bar{x})[\bar{s}]^T (G(\bar{x}))^\dagger DG(\bar{x})[\bar{s}] \right).$$

in contradiction to the conditions in the theorem.  $\square$

## 16.4 Conclusion

The goal of this chapter is an easy and self-contained presentation of optimality conditions for nonlinear semidefinite programs – but the nature of the second order conditions makes it difficult to live up to this goal. We have outlined regularity conditions and contrasted optimality conditions for NLP and NLSDP

in nondegenerate cases. Under standard nondegeneracy assumptions, the tangential cone of an NLP is polyhedral, but the same is generally not true for the tangential cone of an NLSDP. At an optimal solution the Hessian of the Lagrangian of an NLP is positive semidefinite along directions in the critical cone, but this is generally not the case for NLSDPs. In particular, the latter observation has impact on the design of Sequential Semidefinite Programming approaches. A very detailed discussion (also in case that strict complementarity or some regularity conditions are violated) can be found in the monograph by Bonnans and Shapiro [2].

**Acknowledgements** The author is thankful to two anonymous referees whose constructive criticism helped to improve the presentation of the chapter.

## References

1. Alizadeh, F.: Optimization over the positive semi-definite cone: interior-point methods and combinatorial applications. In: Pardalos, P. (ed.) *Advances in Optimization and Parallel Computing*, pp. 1–25. North-Holland, Amsterdam (1992)
2. Bonnans, F., Shapiro, A.: *Perturbation Analysis of Optimization Problems*. Springer-Verlag, New York (2000)
3. Diehl, M., Jarre, F., Vogelbusch, C.H.: Loss of Superlinear Convergence for an SQP-type method with Conic Constraints. *SIAM J. Opt.* **16**(4), 1201–1210 (2006)
4. Fares, B., Noll, D., Apkarian, P.: Robust control via sequential semidefinite programming. *SIAM J. Contr. Opt.* **40**(6), 1791–1820 (2002)
5. Freund, R.W., Jarre, F., Vogelbusch, C.H.: Nonlinear semidefinite programming: sensitivity, convergence, and an application in passive reduced-order modeling. *Math. Prog.* **109**(2-3), 581–611 (2007)
6. Gaile, S., Leugering, G., Stingl, M.: Free Material Optimization for Plates and Shells. To appear in Proceedings of 23rd IFIP TC 7 Conference on System Modelling and Optimization. Cracow, Poland, July 23–27, 2006
7. Kocvara, M., Stingl, M.: Pennon-AMPL User's Guide (Version 1.3). [http://www2.am.uni-erlangen.de/~kocvara/pennon/pennon1\\_3\\_ampl.pdf](http://www2.am.uni-erlangen.de/~kocvara/pennon/pennon1_3_ampl.pdf)
8. Mangasarian, O.L.: *Nonlinear Programming*. McGraw-Hill, New York (1969)
9. Nesterov, J.E.: Semidefinite relaxation and nonconvex quadratic optimization. *Optim. Meth. Software* **9** 141–160 (1998)
10. Nesterov, J.E., Nemirovsky, A.S.: *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*. SIAM, Philadelphia (1994).
11. Nocedal, J., Wright, S.J.: *Numerical Optimization*. Springer, second edition, (2006).
12. Robinson, S.M.: First order conditions for general nonlinear optimization. *SIAM J. Appl. Math.* **30**, 597–607 (1976)
13. Robinson, S.M.: Generalized equations and their solutions, part II: applications to nonlinear programming. *Math. Prog. Study* **19**, 200–221 (1982)
14. Shapiro, A.: First and second order analysis of nonlinear semidefinite programs. *Math. Prog.* **77**, 301–320 (1997)
15. Sonnevend, G.: Applications of analytic centers to feedback control systems. In: Hinrichsen, D., Martensson, B. (eds.) *Control of uncertain systems*, Bremen, June 1989. Birkhäuser (1990)

# Chapter 17

## Recent Progress in Interior-Point Methods: Cutting-Plane Algorithms and Warm Starts

Alexander Engau

### 17.1 Introduction

During the last few decades, interior-point methods for convex programming have made a significant impact on the modern optimization landscape and are now grown to mature and well established areas of mathematical programming. Beginning in the late 1970s, the development of Khachiyan’s ellipsoid method [38] as first polynomial-time algorithm for linear programming (LP) and Karmarkar’s projective-scaling method [37] as first proven polynomial-time interior-point method (IPM) blurred the long-perceived border between linear and nonlinear optimization and revealed that “the great watershed in optimization isn’t between linearity and nonlinearity, but convexity and nonconvexity” [15, 62]. Since then, hundreds of researchers have joined this “interior-point revolution” [69] and contributed numerous new results and trends to the further growing area of conic convex optimization and its applications in a host of different disciplines including optimal control, design, statistics, finance, and many of the other engineering and management sciences. The reader is referred to the various chapters in this handbook providing best evidence for the breadth and depth of research that has evolved in recent years, and more specifically to the review articles [55, 69] for accounts on the historical and technical development of IPMs.

Forming the foundation for the initial development of conic optimization [54] in general and the area of semidefinite programming (SDP) in particular, the study of different classes of IPMs has already been highlighted in the “old” handbook of SDP [68] in which three of four chapters on algorithms [3, 53, 67] give a detailed overview of some of the most important potential-reduction, primal-dual, and path-following IPM variants. The more recent paradigm of self-regularity that has been

---

A. Engau (✉)

Department of Mathematical and Statistical Sciences, University of Colorado Denver,  
Campus Box 170, PO Box 173364, 80217-3364, Denver, CO, USA  
e-mail: [aengau@alumni.clemson.edu](mailto:aengau@alumni.clemson.edu)

developed since then is described, in this new version, in a new Chap. 15 of this Handbook. In addition, algorithmic implementations of some of the most successful methods are now provided in sophisticated and user-friendly optimization software, which are exemplified in several of the software chapters and more comprehensively discussed in the survey by Mittelmann for this new rendition of the Handbook.

While the theory and algorithms of IPMs today are relatively well understood and already widely used in practice, however, their theoretical efficiency and proven polynomial-time complexity does not – and was not expected to – signify ultimate and universal success in solving each corresponding problem instance without remaining challenges or difficulties. Similar to the LP case in which the simplex method continues to be of major importance in practice and often superior especially for specially-structured problems, alternative algorithms also exist for SDP and some of its related variants. Most notably, bundle methods for nonsmooth optimization can be used to solve certain classes of trace-constrained SDPs as equivalent eigenvalue problems, and the reader can find their description in more detail in the fourth chapter on algorithms in the earlier handbook [33] or the slightly newer review article [52]. Different from IPMs whose second-order nature limits their applicability especially for large-scale problems, bundle methods are able to exploit sparsity and special structures and thus perform better on problems whose geometry is sufficiently well understood, for example, SDP relaxations of many combinatorial problems.

Nevertheless, in contrast to the size limitation of general second-order methods, the rich theory of IPMs continues to provide a strong theoretical tool for the analysis and design of new algorithms that typically is unavailable for the majority of other, lower-order methods. In addition – and more importantly at least for the motivation of the present chapter – IPMs can also be computationally advantageous due to their overall robustness when solving problems whose underlying structure or geometry is less well studied or completely unknown. This independence of problem instance and specific application is rapidly gaining significance as SDP approaches and models become more widely used and need to be routinely solved by researchers and practitioners without time or resources to customize a first-order method. Hence, the main objective of this contribution is to review some of the recent enhancements of IPMs with a particular emphasis on cutting-plane algorithms and warm-starting strategies for LP and SDP relaxations of combinatorial optimization problems. Some preliminaries for this discussion are briefly provided in the following Sect. 17.2. Other means of improvements, such as the exploitation of sparsity and parallelization, are omitted here but treated as subjects in some of the other handbook chapters.

The subsequent central discussion then focuses first on various cutting-plane approaches that replace an original, difficult problem with a sequence of approximate and more easily solvable relaxations which are iteratively improved by adding new violated inequalities that remove infeasible iterates until the new solution becomes feasible also for the initial problem. This basic scheme depends on the suitable selection or generation of separating inequalities and possible ways to accelerate the solution of subsequent relaxations. Some new enhancements of methods based

on analytic centers and related approaches are described in Sect. 17.3, which also highlights some of the potential uses of primal-dual interior-point techniques for the early addition and possible removal of cutting-planes to expedite convergence and keep each subproblem at a manageable size, thus reducing the overall computation time of the algorithm. The second topic that received significant attention in recent years includes interior-point warm starts to overcome the inherent challenge that optimal solutions are necessarily non-interior and thus not suited as initial starting points for IPMs. Predominantly studied for LP, we present possible generalizations for SDP in Sect. 17.4. A practical implementation of some of these ideas with representative computational results are briefly discussed in Sect. 17.5, and the concluding Sect. 17.6 offers summarizing remarks and describes current directions of further research.

## 17.2 Preliminaries and Notation

Let  $C \in \mathbb{R}^{n \times n}$  and  $A_i \in \mathbb{R}^{n \times n}$ ,  $i = 1, \dots, m$ , be real symmetric matrices, and  $b \in \mathbb{R}^m$  be a real vector. A semidefinite program in primal-dual standard form is given by

$$\min C \bullet X \text{ s.t. } \mathcal{A}(X) = b, X \succeq 0 \quad (17.1a)$$

$$\max b^T y \text{ s.t. } S = C - \mathcal{A}^T(y) \succeq 0 \quad (17.1b)$$

where  $X \succeq 0$  constrains the matrix  $X \in \mathbb{R}^{n \times n}$  to be symmetric and positive semidefinite,  $y \in \mathbb{R}^m$  is the unrestricted dual variable,  $S \in \mathbb{R}^{n \times n}$  is the symmetric and positive semidefinite dual matrix, and

$$C \bullet X = \text{Tr}(CX) = \sum_{i=1}^n \sum_{j=1}^n C_{ij} X_{ij}, \quad (17.2a)$$

$$\mathcal{A} : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^m : X \mapsto (A_1 \bullet X, \dots, A_m \bullet X)^T, \quad (17.2b)$$

$$\mathcal{A}^T : \mathbb{R}^m \rightarrow \mathbb{R}^{n \times n} : y \mapsto \sum_{i=1}^m y_i A_i. \quad (17.2c)$$

It is easy to see that problem (17.1) includes linear programs as special case if we let  $C = \text{Diag}(c)$  and  $A_i = \text{Diag}(a_i)$  be the diagonal matrices of  $c \in \mathbb{R}^n$  and of the rows of  $A = (a_1, \dots, a_n)^T \in \mathbb{R}^{m \times n}$ , respectively. The problem pair satisfies weak duality

$$C \bullet X - b^T y = X \bullet S \geq 0 \quad (17.3)$$

for every feasible primal-dual iterate  $(X, y, S)$ , and it satisfies strong duality if there exists a strictly feasible (interior) point  $X > 0$  and  $S > 0$ , for which both  $X$  and  $S$  are positive definite. In particular, then  $(X, y, S)$  is optimal if and only if it is feasible and satisfies the complementarity condition  $XS = 0$ , or equivalently, if  $X \bullet S = 0$ . For simplicity, throughout this chapter we assume that strictly feasible points exist.

### 17.2.1 Primal-Dual Interior-Point Methods

The general scheme of most primal-dual methods is to start from an initial, generally feasible or infeasible interior point  $(X^0, y^0, S^0)$  and to compute in each iteration a direction  $(\Delta X^k, \Delta y^k, \Delta S^k)$  and a step length  $\alpha^k > 0$  so that the sequence of iterates

$$(X^{k+1}, y^{k+1}, S^{k+1}) = (X^k, y^k, S^k) + \alpha^k (\Delta X^k, \Delta y^k, \Delta S^k) \quad (17.4)$$

remains interior but converges to an optimal solution of (17.1). The central idea for the computation of directions stems from the concept of the central path that can either be motivated from a perturbed set of first-order optimality conditions of problem (17.1), or derived from the optimality conditions of its primal (or dual) barrier problem

$$\min C \bullet X - \mu \log \det(X) \text{ s.t. } \mathcal{A}(X) = b \quad (X > 0). \quad (17.5)$$

The log-determinant term  $\log \det(X) = \log \prod_{i=1}^n \lambda_i(X) = \sum_{i=1}^n \log \lambda_i(X)$  forms a self-concordant barrier function for the set of symmetric, positive definite matrices with strictly positive eigenvalues  $\lambda_i > 0$ ,  $i = 1, \dots, n$ , and  $\mu > 0$  is a positive barrier parameter. Then substituting  $S = \mu X^{-1}$  in the optimality conditions of problem (17.5), the parametrized solutions  $(X(\mu), y(\mu), S(\mu))$  define the primal-dual central path

$$C = \{(X(\mu), y(\mu), S(\mu)) : X(\mu) > 0, S(\mu) > 0, \mu > 0, \quad (17.6a)$$

$$\mathcal{A}(X(\mu)) = b, \mathcal{A}^T(y(\mu)) + S(\mu) = C, X(\mu)S(\mu) = \mu E\} \quad (17.6b)$$

where  $E \in \mathbb{R}^{n \times n}$  is the identity, and  $(X(\mu), y(\mu), S(\mu))$  is unique for every  $\mu > 0$ .

Path-following IPMs compute iterates in a neighborhood of  $C$  by taking steps into a direction that solves a suitable linearization of the associated Newton system

$$\mathcal{A}(\Delta X) = b - \mathcal{A}(X) \quad (17.7a)$$

$$\mathcal{A}^T(\Delta y) + \Delta S = C - \mathcal{A}^T(y) - S \quad (17.7b)$$

$$X(\Delta S) + (\Delta X)S = \mu E - XS \quad (17.7c)$$

and successively reduce the barrier parameter  $\mu$  toward zero. Because (17.7c) is typically not symmetric, unlike for LP, system (17.7) is generally overdetermined and therefore must be symmetrized. The reader is referred to the old handbook [53, 67] for a discussion of different strategies and the resulting families of search directions. The generic primal-dual path-following IPM outlined above is given in Algorithm 1.

Like every other IPM, primal-dual algorithms usually terminate at approximate solutions within a specified tolerance  $\epsilon$  of a feasible and optimal solution.

**Algorithm 1** Generic Primal-Dual Path-Following IPM for SDP

---

**Input:** Data instance  $(C, \mathcal{A}, b)$ , interior point  $(X^0, y^0, S^0)$ , barrier reduction parameter  $\sigma \in [0, 1]$ , termination tolerance  $\epsilon > 0$ .

**Initialize:** Start iteration counter  $k = 0$ , and set  $\mu^k = \sigma(X^k \bullet S^k)/n$ .

**Until**

$$\max\{\mu^k, \|b - \mathcal{A}(X^k)\|, \|C - \mathcal{A}^T(y^k) - S^k\|\} \leq \epsilon$$

**Do**

- 1. Compute a search direction  $(\Delta X, \Delta y, \Delta S)$  from the linear system (17.7).
- 2. Compute a step size  $\alpha^k > 0$  from a line search along  $(\Delta X, \Delta y, \Delta S)$ .
- 3. Compute  $(X^{k+1}, y^{k+1}, S^{k+1}) = (X^k, y^k, S^k) + \alpha^k(\Delta X, \Delta y, \Delta S)$ .
- 4. Increment  $k = k + 1$ , update  $\mu^k = \sigma(X^k \bullet S^k)/n$ , and repeat.

---

**Output:** Epsilon-approximate solution  $(X^*, y^*, S^*) = (X^k, y^k, S^k)$ .

---

The second parameter  $\sigma$  in Algorithm 1 controls the reduction of the barrier parameter in each iteration and particularly for the class of predictor-corrector methods can be varied between 0 and 1 to result in the best known iteration complexity of  $O(\sqrt{n}L\log(1/\epsilon))$  for any IPM, where  $L$  is used to denote the bit length of the (rational) problem data.

### 17.2.2 Handling of Linear Inequalities

For reference in later sections we consider and briefly discuss two non-standard forms of semidefinite programs using inequality constraints and unrestricted variables in the primal problem, respectively. The first situation arises naturally in cutting-plane methods for combinatorial problems which modify the standard formulation (17.1) by additional linear inequalities on the primal side, and a new associated dual variable

$$\min C \bullet X \quad \text{s.t. } \mathcal{A}(X) = b, \mathcal{P}(X) \geq q, X \geq 0 \quad (17.8a)$$

$$\max b^T y + q^T z \quad \text{s.t. } S = C - \mathcal{A}^T(y) - \mathcal{P}^T(z) \geq 0, z \geq 0. \quad (17.8b)$$

Here  $\mathcal{P}$  and  $\mathcal{P}^T$  are defined analogously to  $\mathcal{A}$  and  $\mathcal{A}^T$  in (17.2b) and (17.2c) using real symmetric  $n \times n$  matrices  $P_i$ ,  $q$  is a real vector, and  $z$  is the additional nonnegative dual variable. We let  $w = \mathcal{P}(X) - q \geq 0$  denote the primal nonnegative slack variable and for every feasible primal-dual iterate  $(X, w, y, z, S)$  obtain a new duality gap of

$$C \bullet X - b^T y - q^T z = X \bullet S + w^T z \geq 0. \quad (17.9)$$

In the central path equations (17.6) and the corresponding augmented Newton system, (17.7a) and (17.7c) stay the same for (17.10a) and (17.10d), (17.7b) is changed to (17.10c), and (17.10b) and (17.10e) account for the new constraint and complementarity

$$\mathcal{A}(\Delta X) = b - \mathcal{A}(X) \quad (17.10a)$$

$$\mathcal{P}(\Delta X) - \Delta w = q - \mathcal{P}(X) + w \quad (17.10b)$$

$$\mathcal{A}^T(\Delta y) + \mathcal{P}^T(\Delta z) + \Delta S = C - \mathcal{A}^T(y) - \mathcal{P}^T(z) - S \quad (17.10c)$$

$$X(\Delta S) + (\Delta X)S = \mu E - XS \quad (17.10d)$$

$$W(\Delta z) + Z(\Delta w) = \mu e - Wz \quad (17.10e)$$

where  $W = \text{Diag}(w)$ ,  $Z = \text{Diag}(z)$ , and  $e = (1, \dots, 1)^T$  is the vector of all ones. In particular, unlike  $X$  and  $S$ , the two diagonal matrices  $W$  and  $Z$  commute and thus do not cause any new computational difficulties other than the often significant increase in size, if the number of inequalities is large. Moreover, whereas all equality constraints are satisfied at optimality and hence active by definition, especially in the context of relaxations many inequalities are likely to remain inactive and can be removed in principle, if known in advance. Typically, the redundancy of a primal constraint is indicated by a zero dual variable, but since IPMs always terminate at approximate solutions that are still interior, the complementarity condition  $WZ = 0$ , or equivalently,  $w_i z_i = 0$  is never satisfied exactly. For the common attempt to fix primal or dual variables to zero, which then can be removed and thereby reduce the problem size, a useful concept is based on the notion of an indicator function [19].

Let  $(X^k, w^k, y^k, z^k, S^k)$  be a sequence of iterates defined by (17.4) and

$$(w^{k+1}, z^{k+1}) = (w^k, z^k) + \alpha^k(\Delta w^k, \Delta z^k) \quad (17.11)$$

that converges to an optimal solution  $(X^*, w^*, y^*, z^*, S^*)$  for problem (17.8). Denote by  $\mathcal{Z}_0^* = \{i : z_i^* = 0\}$  the set of zero dual variables at optimality. An indicator  $\chi$  for  $\mathcal{Z}_0^*$  can be defined as a function that assigns to each  $(w^k, z^k, \Delta w^k, \Delta z^k)$  a vector of extended reals  $\chi(w^k, z^k, \Delta w^k, \Delta z^k)$  for which there exist vectors  $\theta$  and  $\phi$  such that

$$\lim_{k \rightarrow \infty} \chi_i(w^k, z^k, \Delta w^k, \Delta z^k) = \begin{cases} \theta_i & \text{if } i \in \mathcal{Z}_0^* \\ \phi_i & \text{if } i \notin \mathcal{Z}_0^* \end{cases} \quad (17.12)$$

and  $\max \theta_i < \min \phi_i$ . It is said to satisfy a sharp or uniform separation property if

$$\max\{\theta_i : i \in \mathcal{Z}_0^*\} \ll \min\{\phi_i : i \notin \mathcal{Z}_0^*\}, \text{ or if } \theta_i = \theta \text{ and } \phi_i = \phi$$

for some  $\theta$  and  $\phi$  for all  $i \in \mathcal{Z}_0^*$  and  $i \notin \mathcal{Z}_0^*$ , respectively. Common functions include a variable indicator  $\chi^{\text{var}}$ , a primal-dual indicator  $\chi^{\text{prd}}$ , and the Tapia indicator  $\chi^{\text{tap}}$

$$\chi_i^{\text{var}}(w^k, z^k, \Delta w^k, \Delta z^k) = z_i^k, \quad \theta_i^{\text{var}} = \phi_i^{\text{var}} = z_i^*, \quad (17.13a)$$

$$\chi_i^{\text{prd}}(w^k, z^k, \Delta w^k, \Delta z^k) = z_i^k / w_i^k, \quad \theta^{\text{prd}} = 0, \phi^{\text{prd}} = \infty, \quad (17.13b)$$

$$\chi_i^{\text{tap}}(w^k, z^k, \Delta w^k, \Delta z^k) = (z_i^k + \Delta z_i^k) / z_i^k, \quad \theta^{\text{tap}} = 0, \phi^{\text{tap}} = 1. \quad (17.13c)$$

Both primal-dual and Tapia indicator satisfy sharp and uniform separation under strict complementarity, while only the Tapia indicator is scale independent and its identification test  $\chi_i^{\text{tap}} < \theta^{\text{tap}} \Rightarrow z_i^* = 0$  relatively insensitive to the choice of the threshold parameter  $\theta^{\text{tap}}$ . The variable indicator, although the simplest and most popular technique, does not satisfy these properties and thus may not give sufficiently quick and accurate information to improve the performance of an algorithm. A discussion of further properties and indicators is given in the review article [19].

### 17.2.3 Handling of Free Variables

The second non-standard form of relevance for some approaches in our later discussion uses a new vector  $u$  of unconstrained variables in the primal equality constraints

$$\min C \bullet X + d^T u \text{ s.t. } \mathcal{A}(X) + Bu = b, \quad X \geq 0 \quad (17.14a)$$

$$\max b^T y \text{ s.t. } B^T y = d, \quad S = C - \mathcal{A}^T(y) \geq 0 \quad (17.14b)$$

where  $d$  and  $B$  are the primal objective vector and constraint matrix associated with  $u$ , respectively. Although this extended formulation fits easily into the general theoretical framework of primal-dual IPMs, specifically with the same duality gap

$$C \bullet X + d^T u - b^T y = X \bullet S \geq 0 \quad (17.15)$$

as the standard formulation (17.1), some practical consequences affect the computation of search directions  $(\Delta X, \Delta u, \Delta y, \Delta S)$  from the corresponding new Newton system

$$\mathcal{A}(\Delta X) + B\Delta u = b - \mathcal{A}(X) - Bu \quad (17.16a)$$

$$B^T \Delta y = d - B^T y \quad (17.16b)$$

$$\mathcal{A}^T(\Delta y) + \Delta S = C - \mathcal{A}^T(y) - S \quad (17.16c)$$

$$X(\Delta S) + (\Delta X)S = \mu E - XS. \quad (17.16d)$$

Discussed by Anjos and Burer [6], this Newton system remains invertible but now becomes indefinite and thus more difficult to solve in a quick, stable, and accurate manner. Furthermore, the often proposed approach to replace  $u$  by the difference of two nonnegative variables  $u = u^+ - u^-$ ,  $u^+ \geq 0$ ,  $u^- \geq 0$ , and then solve an associated symmetric quasi-definite system has the simultaneous effect that the dual equality  $B^T y = d$  is split into  $B^T y \geq d$  and  $B^T y \leq d$  which can be problematic due to degeneracy, unboundedness of the primal optimal solution set, and loss of an interior for the dual. In consideration of alternative approaches for LP [44] and SDP

[39], they use ideas by Mészáros [44] and provide theoretical and computational support to handle free variables by regularization techniques that perturb equation (17.16b) to

$$B^T \Delta y - \delta \Delta u = d - B^T y \quad (17.17)$$

where  $\delta > 0$  is the positive regularization parameter. This technique results in a definite system that can also maintain structure and sparsity of the matrices  $A_i$  and  $B$ . Although the new search direction may deviate from the original direction by  $O(\delta)$ , the careful choice of the regularization parameter, such that  $\delta^k \|\Delta u^k\| \leq \beta \sigma \mu / 2$  in every iteration for some constant  $\beta > 0$ , guarantees the global convergence of a primal-dual path-following IPM to an optimal solution of the original problem (17.14).

### 17.3 Interior-Point Cutting-Plane Algorithms

The study of cutting-plane methods based on interior-point algorithms has received long-standing attention, and the reader is advised to consult the excellent tutorial chapter by Mitchell [48] for a recent survey and more detailed review of the relevant literature also for more general column-generation schemes, that include the use of IPMs in cutting-plane and cutting-surface methods for LP and SDP together with some related subgradient approaches for nonsmooth optimization. Mitchell was also among the first to explore specifically Karmarkar's algorithm [37] for the solution of LP relaxations of combinatorial problems [45, 51], whereas improvements by SDP relaxations and their solvability with more general IPMs were largely prepared by Alizadeh [1, 2] and since then have led to a rich methodology and applicability of IPMs combined with cutting planes, branch-and-bound, or branch-and-cut. Another useful account in which much of the former material is comprehensively described is the recent, highly recommended survey chapter by Krishnan and Terlaky [42].

Computational studies of IPMs for LP and SDP relaxations with cutting planes have shown encouraging results [7, 34, 46, 49] but also revealed some practical limitations compared to active-set and simplex-based methods in the context of LP, that benefit from their superior warm-start capabilities, and in contrast to bundle methods for SDP that offer better ways to exploit structure and sparsity [24, 32]. Few attempts to design simplex-type methods for SDP were conceptually satisfying but have not yet been practically successful [59]. In particular, unlike simplex or bundle methods so far only IPMs provide theoretical performance guarantees and are shown to run in polynomial time also in the worst case. Besides their importance to prove convergence and analyze the complexity of algorithms, from a practical perspective it has been argued that IPMs can compete with simplex methods if many constraints are added at once, or if it is necessary to stabilize a column-generation process [48]. An early observation by Bixby et al. [14] states that especially in the situation of multiple dual solutions, the tendency of IPMs to produce points in the center of the optimal face typically facilitates the generation of stronger cutting planes than

corresponding simplex-based methods, whose solutions coincide with face vertices that may give much weaker cuts. Consequently, some implementations use IPMs to expedite convergence and gain stability when adding large numbers of inequalities in earlier iterations, and only switch to a simplex-based approach once the process is close to optimality so that only few minor changes remain necessary [14, 50].

Primarily focusing on SDP relaxations, this section continues to describe cutting-plane methods that are based solely on IPMs and designed to exploit their strengths in utilizing interior points and specifically analytic centers for stronger separation, as well as the stability of primal-dual methods for a dynamic mechanism to add and again remove possibly large numbers of inequalities already at intermediate iterates.

### 17.3.1 Basic SDP Relaxation and Cutting Planes

Many SDP relaxations in combinatorial optimization arise from unconstrained or constrained binary quadratic programs in one of three essentially equivalent forms

$$\min \quad u^T Qu + q^T u \quad \text{s.t. } u_i \in \{0, 1\} \quad \forall i \quad (u \in \mathcal{Q}(u)) \quad (17.18a)$$

$$\min \quad v^T Qv + 2c^T v \quad \text{s.t. } v_i \in \{-1, 1\} \quad \forall i \quad (v \in \mathcal{Q}(v)) \quad (17.18b)$$

$$\min \quad x^T Cx \quad \text{s.t. } x_i \in \{-1, 1\} \quad \forall i \quad (x \in \mathcal{Q}(x)). \quad (17.18c)$$

Using the variable transformation  $v = 2u - e$  and suitable definition of the objective vector  $c = Qe + q$ , the binary quadratic  $(0, 1)$ -program (17.18a) is equivalent to the  $(-1, 1)$ -program (17.18b) which can also be written as (17.18c) if increasing its dimension by one, augmenting the matrix  $Q$  with the vector  $c$  in the new row and column,  $C = \begin{pmatrix} Q & c \\ c^T & 0 \end{pmatrix}$ , and introducing an additional variable whose value can be set arbitrarily to either plus or minus 1 [9, 17, 31, 34]. For the converse, it is clear that the last formulation is also subsumed in (17.18b) by letting  $Q = C$  and  $c = 0$ ; furthermore it is well known to be equivalent to the maximum-cut problem if  $C = (W - \text{Diag}(We))/4$ , where  $W$  is the weighted adjacency matrix of the underlying graph. Finally, for each formulation the set  $\mathcal{Q}$  may denote a set of other polyhedral equalities or inequalities.

Starting from the max-cut formulation (17.18c), the typical SDP relaxation follows from the observation that  $x^T Cx = C \bullet xx^T = C \bullet X$ , where  $X = xx^T$  for some  $(-1, 1)$ -vector  $x$  if and only if  $X$  belongs to the cut polytope  $\mathcal{E}^1$ , which is defined as the set of symmetric positive semidefinite matrices that have unit diagonal and are of rank 1. The ellotope  $\mathcal{E}$  relaxes  $\mathcal{E}^1$  by dropping the non-convex rank-1-constraint and can be tightened by additional triangle inequalities that produce the metric polytope  $\mathcal{E}^2$

$$\mathcal{E} = \{X \geq 0 : \text{diag}(X) = e\} \quad \mathcal{E}^2 = \mathcal{E} \cap \{X : X_{ij} + X_{ik} + X_{jk} \geq -1,$$

$$\mathcal{E}^1 = \mathcal{E} \cap \{X : \text{rank}(X) = 1\} \quad X_{ij} - X_{ik} - X_{jk} \geq -1\}$$

**Algorithm 2** Generic Cutting-Plane Method for SDP Relaxations

---

**Input:** Initial SDP relaxation, initial point, termination tolerance  $\epsilon > 0$ .

**Until** the current point is within some  $\epsilon$  of the optimal solution set **do**

- 1. Solve the current relaxation for an (approximate) optimal solution.
- 2. Find or generate one or more inequalities violated by that solution.
- 3. Add the new inequalities as cuts and possibly drop some old ones.

**Output:** Epsilon-approximate solution with a relevant set of inequalities.

---

where  $\mathcal{E}^1 \subseteq \mathcal{E}^2 \subseteq \mathcal{E}$ . In particular, the resulting SDP relaxations over ellipope or metric polytope match either the standard form (17.1) with  $\mathcal{A}(X) = \text{diag}(X)$  and  $b = e$

$$\min C \bullet X \text{ s.t. } \text{diag}(X) = e, X \geq 0 \quad (17.19)$$

or the non-standard form (17.8), respectively. Because the large number of  $O(n^3)$  triangle inequalities in  $\mathcal{E}^2$  cannot be included all at once, however, a relevant subset is typically chosen and added successively within the general framework of a cutting-plane method. Finally, several other yet exponential classes of valid inequalities for the cut polytope  $\mathcal{E}^1$  include the more general hypermetric and odd-cycle inequalities

$$aa^T \bullet X \geq 1 \text{ where } a_i \in \{0, 1\}, \min_{x_j \in \{-1, 1\}} |a^T x| = 1, e^T a \text{ odd} \quad (17.20a)$$

$$X(C \setminus F) - X(F) \leq |C| - 2 \text{ where } C \text{ is cycle, } F \subseteq C, |F| \text{ odd} \quad (17.20b)$$

which are described and analyzed in more detail in a classic monograph by Deza and Laurent [18]. The basic scheme of cutting-plane methods is outlined in Algorithm 2.

Basically independent of the specific algorithm for Step 1, violated cutting planes in Step 2 are often found using complete enumeration especially for triangle inequalities and other polynomial classes of cuts. For exponentially many cutting planes, such as clique or odd-cycle inequalities, other strategies are to extend already known cuts by growing previously selected cliques or cycles, or to use one of several other separation routines or cleverly chosen heuristics. Specific details for Steps 1 and 3, including warm starts and indicators which inequalities to drop, are discussed later.

### 17.3.2 Analytic-Center and Cutting-Surface Methods

Similar in spirit to the original ellipsoid (non-interior-point) method by Khachiyan [38], the basic idea of an analytic-center cutting-plane method (ACCPM) is to generate a sequence of approximate descriptions to some set of interest until finding a point that belongs to this set within some tolerance  $\epsilon$ . Often employed

**Algorithm 3** Generic Analytic-Center Cutting-Plane Method

**Input:** Feasibility instance  $C$ , initial outer approximation, tolerance  $\epsilon > 0$ .

**Until** feasible point is found, or maximum number of iterations is reached **do**

- 1. Compute the (approximate) analytic center of the current relaxation.
- 2. Call the oracle to confirm feasibility or otherwise to obtain a new cut.
- 3. Possibly remove some inequalities or previously added cutting planes.

**Output:** Epsilon-approximate solution in  $C$ , or indication that  $C$  is empty.

---

as initial step in a larger interior-point scheme, these methods can also be used for the purpose of optimization, e.g., if the set  $C$  is chosen as the set of optimal solutions to some LP or SDP relaxation with a large number of inequalities whose direct solution is computationally intractable. The study and analysis of the related algorithms is commonly based on a formulation like the following generic, convex feasibility problem [48]

*Given a convex set  $C \subseteq \mathbb{R}^m$ , either prove that  $C$  is empty or find a point  $y \in \mathbb{R}^m$  such that the Euclidean distance from  $y$  to  $C$  is no greater than  $\epsilon$ .*

In its basic scheme, an ACCPM for problems of the form (17.8) typically starts from an initial outer approximation of the dual feasible region  $\{y \in \mathbb{R}^m : S = C - \mathcal{A}^T(y) \geq 0\}$  and subsequently computes the new analytic center of each current relaxation, that is initially characterized by the corresponding pair of primal and dual barrier problems

$$\min C \bullet X - \log \det(X) \text{ s.t. } \mathcal{A}(X) = 0 \quad (X > 0) \quad (17.21a)$$

$$\max -\log \det(S^{-1}) \text{ s.t. } \mathcal{A}^T(y) + S = C \quad (S > 0). \quad (17.21b)$$

Under the common assumption that the solution set  $C$  is bounded and contained in some box around the origin, the ACCPM algorithm then calls an oracle subroutine that either confirms that the new center belongs to  $C$ , in which case the process is terminated, or returns a separating inequality that is violated by the current point and defines a new cut that contains  $C$  in one of its two halfspaces. This basic scheme is summarized in Algorithm 3 and concludes that the solution set  $C$  is empty if no feasible point could be found after some permissible maximum number of iterations.

For the special case of LP, corresponding algorithms are well studied with a complexity of  $O(m \log(1/\epsilon)^2)$  whose proof largely depends on the use of efficient IPMs for the repeated solution of analytic centers. In particular, ACCPM variants that use approximate or volumetric centers further improve both practically and theoretically, reducing the complexity to  $O(m \log(1/\epsilon))$ . Good reviews that are focused primarily on LP methods are given by Goffin and Vial [27] and Mitchell [47]. The above extension to SDP, where the initial relaxation is defined by a semidefiniteness constraint and linear constraints are added as cutting planes, remains solvable in finitely many iterations that are polynomial in the number of initial constraints and the maximum number of constraints added at each iteration [16, 64, 66]. The use of

linear cut inequalities has also been extended to second-order cuts [57, 58] whose new analytic centers can be recovered within  $O(p \log(p+1))$  Newton steps where  $p$  is the number of cuts added, to semidefinite cuts [56], and in other related ACCPM approaches for LP in combination with column generation and branch-and-price [20] as well as for feasibility problems in more general conic optimization [10, 11].

The extension from adding linear inequalities as cutting planes to second-order or semidefinite cuts as cutting surfaces can be exemplified by the SDP formulation

$$\min C \bullet PVP^T \text{ s.t. } \mathcal{A}(PVP^T) = b, V \succeq 0 \quad (17.22a)$$

$$\max b^T y \text{ s.t. } \mathcal{A}^T(y) + S = C, P^T S P \succeq 0 \quad (17.22b)$$

where the primal matrix  $X = PVP^T$  is constrained by a new variable matrix  $V$  and a matrix  $P$  that simultaneously relaxes positive semidefiniteness in the dual. In particular, if  $\text{rank}(P) = n$ , then this pair of problems is equivalent to the original primal-dual pair (17.1) but otherwise enables to relax the dual by imposing additional restrictions on  $V$ . For example, the LP-like requirement that  $V$  be a diagonal matrix implies that the positive semidefiniteness constraint in the dual problem is relaxed to mere nonnegativity of the diagonal entries of  $P^T S P$ . Similarly, it is possible to impose a block-diagonal structure onto  $V$  which in effect relaxes the dual, because only the corresponding blocks of  $P^T S P$  then need to be positive semidefinite [56].

Other classes of cutting-plane or cutting-surface methods for SDP relaxations specifically over the set of constant-trace matrices  $\mathcal{X} = \{X \geq 0 : E \bullet X = 1\}$ , where  $E$  is the identity and the right-hand-side set to one without loss of generality, arise from the Lagrangean dual and its formulation as equivalent eigenvalue optimization

$$\max_y \Phi(y) = \max_y \min_{X \in \mathcal{X}} b^T y + (C - \mathcal{A}^T(y)) \bullet X \quad (17.23a)$$

$$= \max_y b^T y + \lambda_{\min}(C - \mathcal{A}^T(y)). \quad (17.23b)$$

For the inner minimization subproblem over the set  $\mathcal{X}$  in (17.23a), an optimal solution is given by  $X = uu^T$  where  $u$  is a vector from that eigenspace associated with the smallest (possibly negative) eigenvalue  $\lambda_{\min}(C - \mathcal{A}^T(y))$  of the dual slack matrix  $S = C - \mathcal{A}^T(y)$ . In particular, then the positive semidefiniteness of  $S$  in the original dual is equivalent to the nonsmooth constraint  $\lambda_{\min}(S) \geq 0$  which can be relaxed and successively used to generate semidefinite cutting surfaces of the form  $u^T S u \geq 0$ .

A good survey and further discussion of the above methods and several others is also given by Krishnan and Mitchell [41] who present a unifying framework of several cutting-plane methods for trace-constrained SDPs using the augmented formulation

$$\max y_0 + b^T y - (\gamma/2) \|y - \hat{y}\|^2 \text{ s.t. } P_i^T (C - \mathcal{A}^T(y)) P_i \succeq y_0 E \quad (17.24)$$

where  $y_0$  and  $y$  are the optimization variables,  $\hat{y}$  is the current iterate, and  $\gamma \geq 0$  is a positive parameter that is adjusted dynamically based on the progress of

the algorithm. The variable  $y_0$  corresponds to the primal constant-trace constraint  $E \bullet X = 1$ , and the matrices (or vectors)  $P_i$  are the cuts that are generated in each iteration. Depending on how the bundle  $\mathcal{P} = \{P_1, P_2, \dots\}$  is treated, e.g., aggregated into a single matrix or added as separate constraints, and whether or not the parameter  $\gamma$  is zero or positive, formulation (17.24) reduces to various other known cutting-plane methods including the classical polyhedral cutting-plane method ( $\gamma = 0$ ,  $P_i$  a vector), a primal active-set technique ( $\gamma = 0$ ,  $P_i$  a matrix), or the classical polyhedral or spectral bundle method ( $\gamma > 0$ ,  $P_i$  a vector or matrix, respectively), among others.

### 17.3.3 Approaches Based on Primal-Dual IPMs

The efficiency of IPMs to compute (approximate) optimal solutions or analytic centers in each iteration of the basic cutting-plane scheme in Algorithm 2 or the ACCPM in Algorithm 3 is often countered by their limitation to effectively use previous iterates to warm start or otherwise facilitate the solution of successive relaxations. In addition to the different warm-starting strategies that are described in the next section, this observation also motivated to investigate the effects of adding violated inequalities already much earlier at intermediate iterates, that are on one hand still relatively far from the optimal solution but, on the other hand, sufficiently interior to possibly produce much stronger cuts [34, 49]. In particular, the stability of primal-dual methods may enable to continue the algorithm with only minor adjustments to the current iterate and without the necessity to restart from a new initial point.

The principal success of such approaches based on a primal-dual path-following method for SDP relaxations of binary quadratic programs is first demonstrated by Helmberg and Rendl [34] who distinguish small and large adds at intermediate and final iterates of different rounds, respectively. Small adds allow the early addition of violated inequalities and are compensated by only a minor adjustment that pushes the current iterate back into the interior along the line segment towards the analytic center so that newly added violated inequalities are again satisfied. After every small add typically three regular iterations suffice until a full Newton step occurs which thereby recovers feasibility. After a total of 10 small adds no further cuts are added until an optimal solution of the current relaxation is obtained, which then is followed by a large add, removal of redundant inequalities as indicated by small dual variables, and a full restart of the next relaxation round from a newly chosen initial point.

For the selection of violated cuts and the detection of redundant inequalities, two common criteria are the amount of violation and the value of each associated dual variable, respectively, which essentially coincide with the variable indicators described in Sect. 17.2.2. Fortunately, the drawbacks of this indicator for the detection of zero variables are largely irrelevant for the detection of new inequalities

which are not yet included in the problem and therefore not constrained by barrier terms that prevent their violation. Its lack of sharp and uniform property and its scale dependence are more significant for the removal of unnecessary inequalities, especially at intermediate iterates that inhibit their early recognition because the barrier parameter may still be too large for the dual variables to converge to zero [34]. In the context of LP, Mitchell [46] experienced that the costs of more sophisticated tests for the removal of inequalities outweigh the benefits of the reduction in the relaxation size. For SDP, some encouraging approaches based on the more promising primal-dual and Tapia indicators combined with enhanced strategies for warm-starting are currently explored using a new implementation [23]. These strategies, implementation, and computational results are part of the discussion in the next two sections.

## 17.4 Interior-Point Warm-Starting Strategies

By warm-starting, we mean to exploit information or a known optimal point of some initial optimization problem to accelerate or otherwise facilitate the solution of a closely related problem instance. Of major importance for the solution of successive relaxations in cutting-plane or column-generation approaches as well as branch-and-bound or cut, warm starts play an important role also in many other situations and practical applications. For example, many engineering designs or financial services are subject to frequently changing product specifications or market prices, and one typically anticipates only minor adjustments to accommodate for such changes so that the previous product still provides a good baseline or starting point for the re-optimization process. Repeated changes in the problem data also arise when solving specifically large-scale LPs using Benders or Dantzig–Wolfe decomposition, and especially after few changes it is expected that information from the previously solved problem can be used to simplify the solution of the updated problem instance.

Motivated by the general success of IPMs to provide competitive and at times superior alternatives to simplex algorithms in the case of LP, the problem to warm start IPMs has sparked significant interest and achieved encouraging progress especially during the last few years. Whereas computational savings are still – and will likely remain – inferior to warm starts using a dual simplex method, that benefit from the indirect characterization of solutions in terms of their sets of basic variables which is particularly effective if these sets are not too different, the perception that interior-point warm starts are generally not possible seems rectified but deserves and requires further investigation especially for SDP for which IPM warm starts are frequently without alternative. In the current literature restricted to only LP, each of the following strategies is the author’s often straightforward generalization for SDP; the reader is pointed to the references to compare their original LP formulations.

### 17.4.1 Restoration of Interiority Using Shifted Barriers

It is widely recognized that one of the major challenges in warm-starting IPMs is that optimal or near-optimal solutions are often not suited as initial iterates, and that IPMs tend to perform better when (re-)started from well-centered interior points in proximity to the central path or analytic center rather than close to the boundary of the underlying cone. While several ad-hoc ideas exist to correct or adjust solutions that are close to this boundary, maybe most prominently the concept of an iterate pool that keeps track of intermediate points or approximate analytic centers to not simply choose the last solution but a previous, still sufficiently interior iterate [28], more recent approaches have shown encouraging results based on an earlier idea to relax interiority by one of several modified or shifted-barrier formulations [60].

First considering the SDP in standard form (17.1), the arguably simplest and most intuitive of these formulations is based on the shifted-barrier method by Freund [25]

$$\min C \bullet X - \mu \log \det(X + \mu H) \text{ s.t. } \mathcal{A}(X) = b \quad (X + \mu H > 0) \quad (17.25)$$

where  $H > 0$  is a constant, positive definite shift matrix that relaxes the positive definiteness of  $X$ , and  $\mu$  is the regular barrier parameter that now also controls the amount of relaxation which fully vanishes only as  $\mu$  tends to zero. As discussed in the original reference for LP, the close resemblance of this formulation with the original barrier problem (17.5) facilitates its theoretical analysis which specifically shows that for suitable  $H$  and the proper choice of an initial parameter value  $\mu^0$ , the number of iterations required to achieve  $\mu \leq \epsilon$  is still bounded by  $O(\sqrt{n}L\log(1/\epsilon))$ . Since the choices of both  $H$  and the initial  $\mu^0$  require a priori knowledge of an approximate analytic center of the dual feasible region, however, these results are mostly theoretical and the approach in general not fully satisfactory for practical implementation.

During the last few years, several new approaches have been formulated that adopt the essential idea underlying these shifted-barrier methods to relax the explicit positive semidefiniteness of the primal matrix  $X$  and ensure this necessary condition for feasibility indirectly through some other means. Furthermore, and unlike the first formulation (17.25), especially those methods that relax both  $X$  and the dual matrix  $S'$  within a primal-dual framework have shown good computational performance. Originally formulated for LP, the following SDP formulation is the straightforward extension of the primal-dual penalty method approach by Benson and Shanno [13]

$$\begin{aligned} \min \quad & C \bullet X + D \bullet X' \\ \text{s.t.} \quad & \mathcal{A}(X) = b, U \geq X \geq -X', X' \geq 0 \end{aligned} \quad (17.26a)$$

$$\begin{aligned} \max \quad & b^T y - U \bullet S' \\ \text{s.t.} \quad & \mathcal{A}^T(y) + S = C, D \geq S + S' \geq 0, S' \geq 0 \end{aligned} \quad (17.26b)$$

which introduces two auxiliary non-negative matrices  $X'$  and  $S'$  that relax the positive semidefiniteness of  $X$  and  $S$ , respectively, but are simultaneously penalized in the objective function to again vanish at optimality and thereby confer this condition back onto the original matrices. The associated inequalities require that  $U - X$ , and  $X'$  are non-negative, and that  $X + X'$  is positive semidefinite for the primal, and that  $D - S - S'$ , and  $S'$  are non-negative, and that  $S + S'$  is positive semidefinite for the dual. Hence, for this formulation to be exact it is important that  $D$  and  $U$  are chosen sufficiently large to force  $X'$  and  $S'$  to zero but also to provide admissible upper bounds on the optimal matrices  $X$  and  $S$ . Adaptive strategies to choose and dynamically update these penalties for LP have shown good success and already led to promising computational performance both by itself and combined with additional techniques to identify blocking components of the resulting search direction using sensitivity analysis [30]. In particular, because  $X$  and  $S$  are now unrestricted matrices with free variable entries, a regularized version of this method can conveniently be treated using the framework of Sect. 17.2.3.

A second approach [21] removes the dependency of formulation (17.26) on additional parameters and utilizes the possibility of IPMs to start from infeasible iterates while achieving positive semidefiniteness of  $X$  and  $S$  via their associated slack matrices

$$\min C \bullet X \quad \text{s.t. } \mathcal{A}(X) = b, X - X' = 0, X' \geq 0 \quad (17.27a)$$

$$\max b^T y \quad \text{s.t. } \mathcal{A}^T(y) + S = C, S - S' = 0, S' \geq 0. \quad (17.27b)$$

The redundancy in this formulation, that is always exact and in fact an equivalent reformulation of the original SDP (17.1) in a higher-dimensional space, can be exploited computationally for the efficient solution of the resulting associated Newton system

$$\mathcal{A}(\Delta X) = b - \mathcal{A}(X) \quad (17.28a)$$

$$\mathcal{A}^T(\Delta y) + \Delta S = C - \mathcal{A}^T(y) - S \quad (17.28b)$$

$$\Delta X - \Delta X' = X' - X \quad (17.28c)$$

$$\Delta S - \Delta S' = S' - S \quad (17.28d)$$

$$X'(\Delta S') + (\Delta X')S' = \mu E - X'S' \quad (17.28e)$$

which analogously to (17.7) can be expressed in terms of  $(\Delta X, \Delta y, \Delta S)$  by combining the last three equations into a single, regularized complementary slackness condition

$$X'(\Delta S) + (\Delta X)S' = \mu E - X'S - XS' + X'S'. \quad (17.29)$$

Supported by a theoretical complexity analysis that establishes efficiency of infeasible, primal-dual IPMs for the associated LP formulation [22], this method

also shows encouraging computational performance to warm-start perturbations and SDP relaxations of binary quadratic programs combined with cutting-plane schemes [23].

### 17.4.2 Modified Shifted Barriers in Cutting-Plane Schemes

Similar to Sect. 17.2.2 that extends the standard-form SDP (17.1) by additional inequalities to problems of the form (17.8), we now describe some more specific approaches and show how to modify the previous shifted-barrier methods for warm starts after the addition of cutting planes. Restricting this discussion to linear cuts of the form  $\mathcal{P}(X) \geq q$ , one consequence of including violated inequalities is that the associated slacks  $w = q - \mathcal{P}(X)$  are negative and thus not within the interior of the nonnegative orthant as underlying cone. Possible remedies include either to simply initialize  $w$  to be strictly positive and accept initial infeasibility of the cut inequality, or to attempt to preserve information of the current iterate and – at least temporarily – shift the underlying barrier to also accept an initial point some of whose entries are negative.

First proposed for LP by Mitchell and Todd [51], their alternative approach to deal with the addition of violated cutting planes in an IPM framework is not to relax nonnegativity of the associated slack  $w$  but to modify the system of inequality constraints by an additional variable  $z$  and solve the auxiliary phase-I-type problem

$$\min z \text{ s.t. } \mathcal{A}(X) = b, \mathcal{P}(X) - w + (\beta - \eta)z = q, X \geq 0, (w, z) \geq 0 \quad (17.30)$$

where  $\beta > 0$  is a vector and  $\eta = \mathcal{P}(X^0) - q$  the initial violation at some iterate  $X^0$ , so that  $(X, w, z) = (X^0, \beta, 1)$  is an interior feasible point that fully utilizes the previous iterate. Implementations and computational success with this approach are reported using the primal projective standard-form variant of Karmarkar's algorithm within an LP cutting-plane method applied to matching problems [51], and using the dual affine method combined with a branch-and-bound code for linear ordering [49].

A similar approach that is closely related to the original shifted-barrier methods [25, 60] generalizes the parametrized infeasible-start barrier problem by Freund [26]

$$\min (C + \epsilon(\mathcal{A}^T(y) + \mathcal{P}^T(z) + S - C)) \bullet X - \mu\epsilon \log \det(X) \quad (17.31a)$$

$$\text{s.t. } \mathcal{A}(X) = b, \mathcal{P}(X) - w = q + \epsilon(\mathcal{P}(X) - w - q) (X > 0). \quad (17.31b)$$

This formulation is feasible for  $\epsilon = 1$  independent of the initial value of the slack  $w$ , that therefore can also be set to be strictly positive, and it reduces to the original problem as the parameter  $\epsilon$  and the product  $\epsilon\mu$  are decreased to zero. Unaware of any practical implementation, theoretical results for the LP case show that primal-dual path-following IPMs maintain an iteration complexity of  $O(nL\log(1/\epsilon))$  or  $O(\sqrt{n}L\log(1/\epsilon))$  for suitable infeasible or feasible starting points, respectively [26].

Finally, the two following approaches adjust strategies (17.26) and (17.27) for warm starts after the addition of cutting planes by relaxing nonnegativity of the slack variable  $w$  associated with the violated cut. The primal-dual penalty approach [13]

$$\begin{aligned} \min \quad & C \bullet X + d^T \xi \\ \text{s.t.} \quad & \mathcal{A}(X) = b, \mathcal{P}(X) - w = q, u \geq w \geq -\xi, \xi \geq 0, X \geq 0 \\ \max \quad & b^T y + q^T z - u^T \psi \\ \text{s.t.} \quad & \mathcal{A}^T(y) + \mathcal{P}^T(z) + S = C, d \geq z + \psi \geq 0, \psi \geq 0, S \geq 0 \end{aligned}$$

utilizes auxiliary variables  $\xi$  and  $\psi$  to relax the nonnegativity of  $w$  and its corresponding dual variable  $z$ , and works toward nonnegativity by adding these variables with penalties  $d$  and  $u$  to the two objectives. The alternative slack approach [21, 22]

$$\begin{aligned} \min \quad & C \bullet X \\ \text{s.t.} \quad & \mathcal{A}(X) = b, \mathcal{P}(X) - w = q, w - \xi = 0, \xi \geq 0, X \geq 0 \end{aligned} \tag{17.32a}$$

$$\begin{aligned} \max \quad & b^T y + q^T z \\ \text{s.t.} \quad & \mathcal{A}^T(y) + \mathcal{P}^T(z) + S = C, z - \psi = 0, \psi \geq 0, S \geq 0 \end{aligned} \tag{17.32b}$$

uses  $\xi$  and  $\psi$  slightly differently to regularize the complementarity condition (17.10e) analogously to (17.29). Some results with these approaches are reviewed in Sect. 17.5.

### 17.4.3 Restoration of Feasibility After Perturbations

The problem to restore feasibility and related questions of sensitivity with respect to problem perturbations have led to additional results and techniques in the context of warm-starting IPMs. Although it is principally sufficient to take one Newton step with unit length into the direction  $(\Delta X, \Delta y, \Delta Z)$  obtained from system (17.7) to fully recover primal and dual feasibility, linear approximations of the complementarity equation may cause the resulting iterates  $X + \Delta X$  and  $S + \Delta S$  to become badly centered, or to not anymore be positive (semi)definite. For a pure Newton adjustment

$$\mathcal{A}(\Delta X) = b - \mathcal{A}(X) \tag{17.33a}$$

$$\mathcal{A}^T(\Delta y) + \Delta S = C - \mathcal{A}^T(y) - S \tag{17.33b}$$

$$X(\Delta S) + (\Delta X)S = 0 \tag{17.33c}$$

the analysis by Gondzio and Grothey [29] first provides theoretical bounds on the permissible perturbations drop so that all resulting infeasibilities can be absorbed

by one full Newton step to continue a primal-dual path-following IPM from the new iterate while staying within some predetermined neighborhood of the central path. The applicability of their theoretical results are then demonstrated using an infeasible path-following method and show promising performance also in practice for several large-scale, specially-structured LPs. The relevance of a sensitivity analysis is also highlighted in the unblocking warm-starting approach by the same authors [30].

A similar reoptimization strategy for the restoration of feasibility after problem perturbations is proposed and analyzed by Yildirim and Wright [70] who use primal and dual scaling matrices  $\Sigma$  and  $\Lambda$  for variants of a least-squares adjustment (LSA)

$$\min_{\Delta X} \|\Sigma \Delta X\| \text{ s.t. } \mathcal{A}(X + \Delta X) = b, \quad X + \Delta X \succeq 0 \quad (17.34a)$$

$$\min_{\Delta y, \Delta S} \|\Lambda \Delta S\| \text{ s.t. } \mathcal{A}^T(y + \Delta y) + (S + \Delta S) = C, \quad S + \Delta S \succeq 0. \quad (17.34b)$$

Some specific choices for  $\Sigma$  and  $\Lambda$  include the identity  $E$  for a “plain” LSA, the inverses  $X^{-1}$  and  $S^{-1}$  for a “weighted” LSA, and  $X^{-1/2}S^{1/2}$  and  $X^{1/2}S^{-1/2}$  for a jointly weighted LSA, respectively. The proposed warm-starting strategy is again based on the idea of an iterate pool, for which all intermediate iterates of the initial problem are stored and one of these points is selected as starting point for the perturbed problem instance. Similar to the former approach, several theoretical bounds can be established to estimate the size of perturbations that can be absorbed in only one Newton step by points that are sufficiently interior. The applicability and computational promise of this approach is further demonstrated in a later paper [36].

## 17.5 Implementation and Computational Results

The vast majority of papers cited in this chapter and many of the references therein give computational results that demonstrate the applicability of the various methods and formulations included in our preceding discussion. To exemplify this recently achieved progress of using IPMs for cutting-plane schemes and warm-starting, this particular section first describes a specific implementation developed for one of our own papers [23], that combines a primal-dual interior-point cutting-plane method as described in Sect. 17.3.3 with the slack warm-starting strategy (17.32) and shows its computational success in solving SDP relaxations of maximum cut and the single-row facility layout problem in less time than alternative methods. In addition, we also collect some specific results for warm-starting perturbations of the Netlib LP test problems as reported in four recent papers on warm-starting IPMs [13, 22, 30, 36].

---

**Algorithm 4** Primal-Dual Interior-Point Cutting-Plane Algorithm
 

---

**Input:** Initial SDP relaxation  $(C, \mathcal{A}, b)$  with inequalities  $(\mathcal{P}, q)$ , initial point  $(X^0, y^0, S^0)$ , reduction factor  $\sigma \in [0, 1]$ , termination tolerance  $\epsilon > 0$ , indicator thresholds  $(v_-^{\text{var}}, v_+^{\text{var}}, v_-^{\text{tap}}, v_+^{\text{tap}}, v^{\text{prd}})$ , update frequency  $\kappa$ .

**Initialize:** Set  $k = 0$ ,  $I^k = \emptyset$ ,  $\mu^k = \sigma(X^k \bullet S^k)/n$ ,  $v^k = v^{k-1} = q - \mathcal{P}(X^k)$ .

**Until**

$$\max \left\{ \mu^k, \|b - \mathcal{A}(X^k)\|, \|C - \mathcal{A}^T(y^k) - \mathcal{P}_{I^k}^T(z^k) - S^k\|, v_i^k : i \in I \right\} \leq \epsilon$$

**Do**

1. Set  $I^{k+1} = \begin{cases} I^k \setminus \{i \in I^k : v_i^k \leq \min\{v_-^{\text{var}}, v_-^{\text{tap}}, v_i^{k-1}, v^{\text{prd}}, z_i^k\}\} \\ \cup \{i \notin I^k : v_i^k \geq \min\{v_+^{\text{var}}, v_+^{\text{tap}}, v_i^{k-1}\}\} \text{ if } k \equiv 0 \pmod{\kappa}, \\ I^k \text{ otherwise.} \end{cases}$
2. For  $i \in I^{k+1} \setminus I^k$  add new slacks  $(w_i^k, z_i^k, \xi_i^k, \phi_i^k) = (v_i^k, 0, \sqrt{\mu^k}, \sqrt{\mu^k})$ .
3. Get  $(\Delta X, \Delta w, \Delta y, \Delta z, \Delta S, \Delta \xi, \Delta \psi)$  from the Newton system to (17.32).
4. Get step size  $\alpha^k > 0$  and  $(X^{k+1}, w^{k+1}, y^{k+1}, z^{k+1}, S^{k+1}, \xi^{k+1}, \psi^{k+1})$ .
5. Increment  $k = k + 1$ , update  $\mu^k$ , set  $v^k = q - \mathcal{P}(X^k)$ , and repeat.

**Output:** Epsilon-approximate solution  $(X^*, y^*, S^*) = (X^k, y^k, S^k)$  with a relevant set of inequalities  $I^* = I^k$  that are active at optimality.

---

### 17.5.1 An Interior-Point Cutting-Plane Implementation

The new interior-point cutting-plane method outlined in Algorithm 4 combines several of the previous approaches and ideas for solving SDP relaxations of the form (17.8) by integrating a cutting-plane scheme into the infeasible primal-dual path-following IPM in Algorithm 1. In particular, this method does not require to solve successive relaxations to optimality but adds and removes cuts dynamically at intermediate iterates, using some of the feasibility indicators (17.13) in Sect. 17.2.2 applied to the cut violation  $v = q - \mathcal{P}(X)$  and the shifted-barrier warm-starting technique (17.32) for the initialization of the associated primal-dual slacks  $w$  and  $z$ . The resulting algorithm basically works like a regular IPM that takes steps in different spaces in different iterations  $k$  based on the set of inequalities  $I^k \subseteq I$  currently added to the problem.

In Step 1, indicator functions are applied to the cut violations  $v_i = q_i - P_i \bullet X$  to gain information which inequalities remain inactive ( $v_i < 0$ ), tend to become active ( $v_i = 0$ ), or are violated ( $v_i > 0$ ). New cuts are added aggressively, if either the variable or Tapia indicator show a cut violation, and again removed more conservatively only if their violation is eliminated and their redundancy suggested by both the Tapia and primal-dual indicator. Although this cautious strategy may result in keeping too many inequalities in the problem and still offers room for further improvement, it seems to guarantee that no single inequalities is repeatedly removed and re-added which otherwise be prevented by permanently enforcing an inequality after some finite number of swaps. Unlike the discussion in Sect. 17.2.2, it should also be observed that indicators are not directly applied to the primal-dual

variables  $w$  and  $z$ , which are converted into free variables by introducing the auxiliary slacks  $\xi$  and  $\psi$  as shown in (17.32) and thus handled using the techniques in Sect. 17.2.3.

The initialization of these new variables in Step 2 is motivated by the theoretical analysis and computational experience with the slack warm-starting approach that these seemingly redundant slacks can mitigate the impact of violated cuts and are sufficiently decoupled from the other constraints to allow much smaller initial values than otherwise necessary when initializing the original variables  $(w, z) \gg 0$ . The particular values in Algorithm 4 are chosen so to preserve feasibility of all the original constraints and to continue the algorithm with the same barrier parameter value for the computation of the new search direction in Step 3 as obtained from the previous iterate. The last three steps are also essentially identical to those in Algorithm 1, and without any other changes the current implementation simply removes the associated variables  $(w_i, z_i, \xi_i, \psi_i)$  whenever dropping a previously added inequality  $i \in I$ .

Finally, for some increased flexibility when testing the method, the frequency parameter  $\kappa$  in Algorithm 4 can be chosen so to add new cuts before every Newton iteration (if  $\kappa = 1$ ) or to allow some intermediate steps in which the current set of cuts remains unchanged (if  $\kappa > 1$ ). In particular, better computational results are typically achieved for  $\kappa = 2$ , i.e., when updating the set  $I$  every other iteration. Thus different from the approach by Helmberg and Rendl [34] that is described in more detail in Sect. 17.3.3 and only adds cuts whenever feasibility is fully established, this new implementation also makes use of the computational efficiency and stability of infeasible IPMs that are advantageously exploited by the slack warm-starting approach to more quickly accommodate violated inequalities and handle the resulting infeasible iterates without any additional correction steps or the need to restart.

### 17.5.2 Results on Maximum Cut and Facility Layout

Using an implementation of Algorithm 4 based on the primal-dual path-following IPM in the software package SDPT3 [65], Table 17.1 shows representative results for solving SDP relaxations of max-cut instances (17.19) using triangle inequalities as cuts

$$\min C \bullet X \text{ s.t. } \text{diag}(X) = e, X \succeq 0 \quad (17.35a)$$

$$X_{ij} + X_{jk} + X_{ik} \geq -1 \text{ for } i < j < k \quad (17.35b)$$

$$X_{ij} - X_{jk} - X_{ik} \geq -1 \text{ for distinct } (i, j, k). \quad (17.35c)$$

All test instances were generated using RUDY [61] and include small and medium-sized random graphs of varying density, and 2D and 3D spin glasses with Gaussian or uniformly distributed  $\pm 1$  bonds. For each problem type and size the results are

**Table 17.1** Representative results for RUDY-Generated max-cut instances

Problem(size)	Dynamic cuts (Algorithm 4)				Static cuts	
	Min	Cuts	Max	cpu time	cpu time	$\gamma$
Rand100dens50	1488	1659	1832	61.44	80.85	0.76
Rand100dens90	1777	2097	2867	131.67	172.67	0.76
Rand200dens50	2570	2928	3297	316.05	457.06	0.69
Rand200dens90	2815	2977	3166	310.96	435.42	0.71
Spin2Dgauss100	2409	3466	4981	484.51	643.64	0.75
Spin2Dpm100	1764	2524	3114	271.47	444.64	0.61
Spin2Dgauss225	2145	2799	3436	483.02	558.54	0.86
Spin2Dpm225	2361	3028	4223	243.61	285.17	0.85
Spin3Dgauss125	3275	4133	4909	939.03	1143.50	0.82
Spin3Dpm125	4152	4303	4430	362.47	337.07	1.08
Spin3Dgauss216	3109	5322	6686	2166.37	2284.27	0.95
Spin3Dpm216	3875	4738	5582	456.10	485.38	0.94

averaged over 10 instances, and the cpu times (in seconds) are used to compute a competitiveness ratio  $\gamma$  of the dynamic detection of cuts in Algorithm 4 over the direct solution of the final relaxation with that static subset of cuts that remains active at optimality and for which the dual variables and primal-dual indicators are bounded away from zero. Inequalities were checked every other iteration ( $\kappa = 2$  in Algorithm 4), and the number of new cuts was restricted to at most 200 at a time.

The clear tendency of finding relevant inequalities and an optimal solution faster than the “static-cut” approach that basically assumes an oracle and only solves the final yet immediately much larger relaxation can also be confirmed on SDP relaxations of the single-row facility layout problem (SRFLP) which is further highlighted by Anjos and Liers in Chap. 29. In its standard formulation, each SRFLP instance consists of  $n$  one-dimensional facilities with given positive lengths  $\ell_i$ ,  $i = 1, \dots, n$ , and pairwise weights  $c_{ij}$  with the objective to arrange the facilities so as to minimize the total weighted sum of their center-to-center distances. In particular, if all the facilities have the same length, then SRFLP becomes an instance of the linear ordering problem which is itself a special case of the quadratic assignment problem also showing that SRFLP is NP-hard, in general. Formulated as a constrained binary quadratic program, the following gives the SDP relaxation by Anjos and Yen [8] which again uses the triangle inequalities as cuts

$$\begin{aligned}
& \min \sum_{i < j < k} (c_{ik}\ell_j X_{ij,jk} - c_{jk}\ell_i X_{ij,ik} - c_{ij}\ell_k X_{ik,jk}) \\
& \text{s.t. } \sum_{k \neq i,j} (X_{ij,jk} - X_{ij,ik} - X_{ik,jk}) = -(n-2), \quad \text{diag}(X) = e, \quad X \geq 0, \\
& \quad X_{i_1 i_2, j_1 j_2} + X_{i_1 i_2, k_1 k_2} + X_{j_1 j_2, k_1 k_2} \geq -1 \text{ for ordered } (i_1 i_2, j_1 j_2, k_1 k_2), \\
& \quad X_{i_1 i_2, j_1 j_2} - X_{i_1 i_2, k_1 k_2} - X_{j_1 j_2, k_1 k_2} \geq -1 \text{ for distinct } (i_1 i_2, j_1 j_2, k_1 k_2).
\end{aligned}$$

**Table 17.2** Representative results for single-row facility layout instances

Contributing authors	Dynamic cuts			Static cuts	
	Facilities	Cuts	cpu time	cpu time	$\gamma$
Beginh and Hansen [12]	4	60	0.1	0.1	0.87
Love and Wong [43]	5	317	1.6	3.6	0.45
Heragu and Kusiak [35]	6	758	9.3	20.2	0.46
Heragu and Kusiak [35]	7	801	3.3	5.1	0.65
Simmons [63]	8	1494	16.7	26.2	0.64
Simmons [63]	9	2068	51.0	74.0	0.69
Heragu and Kusiak [35]	12	2800	112.3	135.2	0.83
Heragu and Kusiak [35]	15	5503	957.8	1269.7	0.75
Amaral [4]	15	6361	1875.3	2081.5	0.90
Amaral [5]	17	6581	1811.7	2087.7	0.87
Heragu and Kusiak [35]	20	8000	2008.2	2173.9	0.92
Anjos and Vannelli [7]	25	8199	3708.5	4313.8	0.86

Using several small and medium-sized instances from the SRFLP library [40], the representative results in Table 17.2 are obtained and interpreted analogously to those in Table 17.1. The full result list and discussion is contained in the original paper [23].

### 17.5.3 Warm-Starting Results After Perturbations

The third set of results in Table 17.3 is collected from three warm-starting approaches [13, 21, 30] that report numerical results specifically for LP instances of the Netlib suite after data perturbations. Although only those instances are included for which results were available for all three approaches, the full list looks very similar [22]. Results with the methods in Sect. 17.4.3 are also prepared by John and Yildirim [36] on the same instances, but presented in a different form and thus not repeated here.

For each method, the results in Table 17.3 show the average number of iterations required to solve randomly perturbed instances of every original Netlib problem first using a warm start from the known optimal solution of its unperturbed problem instance, and then again from a cold start using the default initial point specified by the chosen optimization solver. Although using the same perturbation scheme that is explained in each of the relevant papers, the observed differences between the results obtained by each of these three methods are more likely a consequence of the randomly perturbed instances and the specific implementation of their underlying algorithms rather than an indication as to whether a single approach performs better than the two others. Hence without conclusion about a “winner” among these three approaches, however, these results demonstrate that warm-starting IPMs is possible in general and offers reductions in the number of iterations of over 50% on average.

**Table 17.3** Representative Results for Warm-Starting LP Perturbations

Problem	Penalty approach [13]			Unblocking approach [30]			Slack approach [21]		
	Warm	Cold	$\gamma$	Warm	Cold	$\gamma$	Warm	Cold	$\gamma$
adlittle	6.7	17.8	0.38	6.0	10.4	0.57	7.1	14.9	0.48
afiro	3.6	12.0	0.30	4.5	10.2	0.44	5.0	10.6	0.47
agg2	10.2	28.7	0.34	4.6	16.2	0.29	6.8	25.1	0.26
agg3	8.7	24.7	0.35	5.5	15.8	0.35	7.0	23.9	0.29
bandm	10.7	19.2	0.56	6.2	13.7	0.45	8.4	22.8	0.37
beaconfd	5.0	14.0	0.36	4.4	10.2	0.43	4.4	15.3	0.29
blend	6.5	16.0	0.41	5.3	9.1	0.59	6.7	14.3	0.47
degen2	7.1	17.1	0.41	24.0	27.2	0.77	10.8	19.2	0.52
e226	18.4	23.5	0.78	7.8	15.4	0.50	7.4	22.9	0.33
grow15	15.3	24.6	0.62	8.3	16.8	0.48	8.9	24.5	0.34
grow7	11.5	23.5	0.48	9.7	17.0	0.53	7.9	22.5	0.33
israel	9.7	29.4	0.33	4.9	20.4	0.24	5.0	25.9	0.19
kb2	7.4	16.9	0.43	6.4	17.8	0.36	6.1	18.1	0.34
sc105	8.2	14.6	0.56	6.5	11.8	0.56	6.9	12.7	0.54
sc205	11.2	17.5	0.64	8.7	12.3	0.70	7.8	15.3	0.50
sc50a	5.6	13.4	0.42	4.7	11.0	0.42	5.8	11.4	0.51
sc50b	4.2	12.1	0.35	5.7	10.3	0.54	5.7	10.1	0.56
scagr25	11.5	29.2	0.39	4.9	12.2	0.40	7.6	19.2	0.39
scagr7	18.1	25.9	0.70	4.4	10.0	0.44	5.9	17.9	0.33
scsd1	7.6	14.9	0.50	5.8	9.5	0.61	7.4	12.9	0.57
sctap1	12.9	21.0	0.61	6.2	15.4	0.40	8.2	21.8	0.38
share1b	18.2	41.1	0.44	6.5	21.5	0.30	5.8	29.9	0.19
share2b	7.6	15.8	0.48	5.9	9.2	0.64	6.0	16.9	0.35
stocfor1	4.8	17.5	0.27	5.2	13.1	0.40	3.7	17.0	0.22

## 17.6 Conclusion

We reviewed some recent progress of using interior-point algorithms in the context of cutting-plane methods and semidefinite relaxations of combinatorial optimization problems. Today, IPMs are well understood in theory and already widely used in practice, but their further improvement especially for large-scale and specially-structured problems remains a central topic of active research. The second-order nature of IPMs guarantees their exceptional overall stability but often restricts the size of problems that can be handled practically. This challenge gains relevance as LP relaxations nowadays are often replaced by SDP relaxations to provide tighter bounds for many important applications. In particular, in the last few years analytic-center cutting-plane methods for convex feasibility problems have been extended from sets with polyhedral descriptions to sets characterized by positive semidefiniteness constraints and linear matrix inequalities. Moreover, the well-studied approach to tighten successive descriptions using linear inequalities is now enhanced by second-order and semidefinite cuts, and further generalizations to conic optimization remain under current investigation. Alternative approaches based

on primal-dual methods exploit the stability of IPMs for the dynamic addition and removal of cuts at intermediate iterates and are of interest if combined with indicator functions and enhanced strategies for warm-starting after the addition of cuts or after data perturbations. Computational results demonstrate the potential of these approaches.

We conclude with some ideas to promote further progress in this area. The question whether it is possible to also define good second-order cone relaxations for binary quadratic programs that could compromise between the solvability of LP relaxations and the better bounds of SDP relaxations together with the development of new or improved solution strategies using more efficient IPMs or alternative methods seems a promising avenue with only few current contributions. Moreover, the combination of interior-point/simplex schemes that seem to perform very well in the case of LP are currently unavailable for SDP unless significant progress can be made to also design practical simplex-type approaches for SDP and conic optimization in general. The other possibility to develop better warm-starting approaches for SDP are prepared in this chapter, but it is important to reemphasize that these formulations are extended from LP and that their computational impact currently remains unexplored for the addition of nonlinear cuts and for general perturbations of SDPs. Finally, the development of a meaningful suite of reoptimization test problems and a common testing environment for the different approaches remains another urgent necessity to gain further insight and stimulate or accelerate new ideas and progress.

## References

1. Alizadeh, F.: Combinatorial optimization with interior point methods and semidefinite matrices. PhD thesis, University of Minnesota (1991)
2. Alizadeh, F.: Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optim.* **5**(1), 13–51 (1995)
3. Alizadeh, F., Schmieta, S.: Symmetric cones, potential reduction methods and word-by-word extensions. In: Internat. Ser. Oper. Res. Management Sci., Handbook of semidefinite programming, vol. 27, pp. 195–233. Kluwer Acad. Publ., Boston, MA (2000)
4. Amaral, A.R.S.: On the exact solution of a facility layout problem. *European J. Oper. Res.* **173**(2), 508–518 (2006)
5. Amaral, A.R.S.: An exact approach to the one-dimensional facility layout problem. *Oper. Res.* **56**(4), 1026–1033 (2008)
6. Anjos, M.F., Burer, S.: On handling free variables in interior-point methods for conic linear optimization. *SIAM J. Optim.* **18**(4), 1310–1325 (2007)
7. Anjos, M.F., Vannelli, A.: Computing globally optimal solutions for single-row layout problems using semidefinite programming and cutting planes. *INFORMS J. Comput.* **20**(4), 611–617 (2008)
8. Anjos M.F., Yen, G.: Provably near-optimal solutions for very large single-row facility layout problems. *Optimization Methods Software* **24**(4-5), 805–817 (2009)
9. Barahona, F., Jünger, M., Reinelt, G.: Experiments in quadratic 0-1 programming. *Math. Programming* **44**(2, (Ser. A)), 127–137 (1989)

10. Basescu, V.L.: An analytic center cutting plane method in conic programming. PhD thesis, Rensselaer Polytechnic Institute (2003)
11. Basescu, V.L., Mitchell, J.E.: An analytic center cutting plane approach for conic programming. *Math. Oper. Res.* **33**(3), 529–551 (2008)
12. Beghin-Picavet, M., Hansen, P.: Deux problèmes d'affectation non linéaires. *RAIRO Recherche Opérationnelle* **16**(3), 263–276 (1982)
13. Benson, H.Y., Shanno, D.F.: An exact primal-dual penalty method approach to warmstarting interior-point methods for linear programming. *Comput. Optim. Appl.* **38**(3), 371–399 (2007)
14. Bixby, R.E., Gregory, J.W., Lustig, I.J., Marsten, R.E., Shanno, D.F.: Very large-scale linear programming: a case study in combining interior point and simplex methods. *Oper. Res.* **40**(5), 885–897, (1992)
15. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge University Press, Cambridge (2004)
16. Chua, S.K., Toh, K.C., Zhao, G.Y.: Analytic center cutting-plane method with deep cuts for semidefinite feasibility problems. *J. Optim. Theory Appl.* **123**(2), 291–318 (2004)
17. De Simone, C.: The cut polytope and the Boolean quadric polytope. *Discrete Math.* **79**(1), 71–75 (1989)
18. Deza, M.M., Laurent, M.: Geometry of cuts and metrics. In: Algorithms and Combinatorics, vol. 15. Springer-Verlag, Berlin (1997)
19. El-Bakry, A.S., Tapia, R.A., Zhang, Y.: A study of indicators for identifying zero variables in interior-point methods. *SIAM Rev.* **36**(1), 45–72 (1994)
20. Elhedhli, S., Goffin, J.-L.: The integration of an interior-point cutting plane method within a branch-and-price algorithm. *Math. Program.* **100**(2, Ser. A), 267–294 (2004)
21. Engau, A., Anjos, M.F., Vannelli, A.: A primal-dual slack approach to warmstarting interior-point methods for linear programming. In: Chinneck, J.W., Kristjansson, B., Saltzman, M.J. (eds.) Operations Research and Cyber-Infrastructure, pp. 195–217, Springer (2009)
22. Engau, A., Anjos, M.F., Vannelli, A.: On interior-point warmstarts for linear and combinatorial optimization. *SIAM J. Optim.* **20**(4), 1828–1861 (2010)
23. Engau, A., Anjos, M.F., Vannelli, A.: On handling cutting planes in interior-point methods for semidefinite relaxations of binary quadratic optimization problems. Technical Report, University of Waterloo (2010). To appear in *Optim. Methods Softw.*
24. Fischer, I., Gruber, G., Rendl, F., Sotirov, R.: Computational experience with a bundle approach for semidefinite cutting plane relaxations of Max-Cut and equipartition. *Math. Program.* **105**(2-3, Ser. B), 451–469 (2006)
25. Freund, R.M.: Theoretical efficiency of a shifted-barrier-function algorithm for linear programming. *Linear Algebra Appl.* **152**, 19–41 (1991)
26. Freund, R.M.: An infeasible-start algorithm for linear programming whose complexity depends on the distance from the starting point to the optimal solution. *Ann. Oper. Res.* **62**, 29–57 (1996)
27. Goffin, J.-L., Vial, J.-P.: Convex nondifferentiable optimization: a survey focused on the analytic center cutting plane method. *Optim. Methods Softw.* **17**(5), 805–867 (2002)
28. Gondzio, J.: Warm start of the primal-dual method applied in the cutting-plane scheme. *Math. Programming* **83**(1, Ser. A), 125–143 (1998)
29. Gondzio, J., Grothey, A.: Reoptimization with the primal-dual interior point method. *SIAM J. Optim.* **13**(3), 842–864 (2003)
30. Gondzio, J., Grothey, A.: A new unblocking technique to warmstart interior point methods based on sensitivity analysis. *SIAM J. Optim.* **19**(3), 1184–1210 (2008)
31. Hammer, P.L.: Some network flow problems solved with pseudo-Boolean programming. *Operations Res.* **13**, 388–399 (1965)
32. Helmberg, C.: A cutting plane algorithm for large scale semidefinite relaxations. In: The sharpest cut, pp. 233–256, SIAM, Philadelphia, PA (2004)
33. Helmberg, C., Oustry, F.: Bundle methods to minimize the maximum eigenvalue function. In: Internat. Ser. Oper. Res. Management Sci., Handbook of semidefinite programming, vol. 27, pp. 307–337. Kluwer Acad. Publ., Boston, MA (2000)

34. Helmberg, C., Rendl, F.: Solving quadratic (0, 1)-problems by semidefinite programs and cutting planes. *Math. Programming* **82**(3, Ser. A), 291–315 (1998)
35. Heragu S.S., Kusiak, A.: Efficient models for the facility layout problem. *European J. Oper. Res.* **53**(1), 1–13 (1991)
36. John, E., Yildirim, E.A.: Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimension. *Comput. Optim. Appl.* **41**(2), 151–183 (2008)
37. Karmarkar, N.: A new polynomial-time algorithm for linear programming. *Combinatorica* **4**(4), 373–395 (1984)
38. Khachiyan, L.G.: A polynomial algorithm in linear programming. *Dokl. Akad. Nauk SSSR* **244**(5), 1093–1096 (1979)
39. Kobayashi, K., Nakata, K., Kojima, M.: A conversion of an SDP having free variables into the standard form SDP. *Comput. Optim. Appl.* **36**(2–3), 289–307 (2007)
40. Kong, C., Anjos, M.F.: Facility Layout Problem (FLP) Database Library. Available at <http://flplib.uwaterloo.ca/>
41. Krishnan, K., Mitchell, J.E.: A unifying framework for several cutting plane methods for semidefinite programming. *Optim. Methods Softw.* **21**(1), 57–74 (2006)
42. Krishnan, K., Terlaky, T.: Interior point and semidefinite approaches in combinatorial optimization. In: GERAD 25th Anniv. Ser., Graph theory and combinatorial optimization, vol. 8, pp. 101–157. Springer, New York (2005)
43. Love, R., Wong, J.: On solving a one-dimensional space allocation problem with integer programming. *INFOR* **14**(2), 139–143 (1976)
44. Mészáros, C.: On free variables in interior point methods. *Optim. Methods Softw.* **9**(1–3), 121–139 (1998)
45. Mitchell, J.E.: Karmarkar's algorithm and combinatorial optimization problems. PhD thesis, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY (1988)
46. Mitchell, J.E.: Computational experience with an interior point cutting plane algorithm. *SIAM J. Optim.* **10**(4), 1212–1227 (2000)
47. Mitchell, J.E.: Polynomial interior point cutting plane methods. *Optim. Methods Softw.* **18**(5), 507–534 (2003)
48. Mitchell, J.E.: Cutting plane methods and subgradient methods. In: *Tutorials in Operations Research* (M. Oskoorouchi, ed.), Chap. 2, pp. 34–61. INFORMS (2009)
49. Mitchell, J.E., Borchers, B.: Solving real-world linear ordering problems using a primal-dual interior point cutting plane method. *Ann. Oper. Res.* **62**, 253–276 (1996)
50. Mitchell, J.E., Borchers, B.: Solving linear ordering problems with a combined interior point/simplex cutting plane algorithm. In: *Appl. Optim.*, High performance optimization, vol. 33, pp. 349–366. Kluwer Acad. Publ., Dordrecht (2000)
51. Mitchell, J.E., Todd, M.J.: Solving combinatorial optimization problems using Karmarkar's algorithm. *Math. Programming* **56**(3, Ser. A), 245–284 (1992)
52. Monteiro, R.D.C.: First- and second-order methods for semidefinite programming. *Math. Program.* **97**(1–2, Ser. B), 209–244 (2003)
53. Monteiro, R., Todd, M.: Path-following methods. In: *Internat. Ser. Oper. Res. Management Sci.*, Handbook of semidefinite programming, vol. 27, pp. 267–306. Kluwer Acad. Publ., Boston, MA (2000)
54. Nesterov, Y., Nemirovskii, A.: Interior-point polynomial algorithms in convex programming. In: *SIAM Studies in Applied Mathematics*, vol. 13. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA (1994)
55. Nemirovski, A.S., Todd, M.J.: Interior-point methods for optimization. *Acta Numer.* **17**, 191–234 (2008)
56. Oskoorouchi, M.R., Goffin, J.-L.: The analytic center cutting plane method with semidefinite cuts. *SIAM J. Optim.* **13**(4), 1029–1053 (2003)
57. Oskoorouchi, M.R., Goffin, J.-L.: An interior point cutting plane method for the convex feasibility problem with second-order cone inequalities. *Math. Oper. Res.* **30**(1), 127–149 (2005)

58. Oskoorouchi, M.R., Mitchell, J.E.: A second-order cone cutting surface method: complexity and application. *Comput. Optim. Appl.* **43**(3), 379–409 (2009)
59. Pataki, G.: Cone-LP's and semidefinite programs: geometry and a simplex-type method. In: *Lecture Notes in Comput. Sci., Integer programming and combinatorial optimization* (Vancouver, BC, 1996), vol. 1084, pp. 162–174. Springer, Berlin (1996)
60. Polyak, R.: Modified barrier functions (theory and methods). *Math. Programming* **54**(2, Ser. A), 177–222 (1992)
61. Rinaldi, G.: Rudy: a rudimentary graph generator. Available at [http://www-user.tu-chemnitz.de/~helmberg/rudy.tar.gz\(1998\)](http://www-user.tu-chemnitz.de/~helmberg/rudy.tar.gz(1998))
62. Rockafellar, R.T.: Lagrange multipliers and optimality. *SIAM Rev.* **35**(2), 183–238 (1993)
63. Simmons, D.M.: One-dimensional space allocation: An ordering algorithm. *Operations Res.* **17**, 812–826 (1969)
64. Sun, J., Toh, K.-C., Zhao, G.: An analytic center cutting plane method for semidefinite feasibility problems. *Math. Oper. Res.* **27**(2), 332–346 (2002)
65. Toh, K.-C., Todd, M.J., Tütüncü, R.H.: SDPT3—a MATLAB software package for semidefinite programming, version 1.3. *Optim. Methods Softw.* **11/12**(1-4), 545–581 (1999)
66. Toh, K.-C., Zhao, G., Sun, J.: A multiple-cut analytic center cutting plane method for semidefinite feasibility problems. *SIAM J. Optim.* **12**(4), 1126–1146 (2002)
67. Tuncel, L.: Potential reduction and primal-dual methods. In: *Internat. Ser. Oper. Res. Management Sci., Handbook of semidefinite programming*, vol. 27, pp. 235–265. Kluwer Acad. Publ., Boston, MA (2000)
68. Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.): *Handbook of semidefinite programming. International Series in Operations Research & Management Science*, vol. 27. Kluwer Academic Publishers, Boston, MA (2000)
69. Wright, M.H.: The interior-point revolution in optimization: history, recent developments, and lasting consequences. *Bull. Amer. Math. Soc. (N.S.)* **42**(1), 39–56 (2005)
70. Yildirim, E.A., Wright, S.J.: Warm-start strategies in interior-point methods for linear programming. *SIAM J. Optim.* **12**(3), 782–810 (2002)

# Chapter 18

## Exploiting Sparsity in SDP Relaxation of Polynomial Optimization Problems

Sunyoung Kim and Masakazu Kojima

### 18.1 Introduction

Polynomial optimization problems (POP) are nonlinear optimization problems whose objective and constraint functions involve only polynomials. They, however, provide a general framework to represent various application problems in science and engineering. In particular, POPs include quadratic optimization problems with or without 0-1 constraints on their variables. POPs are nonconvex in general, and they have served as a unified mathematical model for the study of global optimization of nonconvex continuous and discrete optimization problems. See the book [24] and the references therein. A POP is described as

$$\text{minimize } f_0(x) \text{ subject to } f_k(x) \geq 0 \ (k = 1, 2, \dots, m), \quad (18.1)$$

where  $f_k$  denotes a polynomial in  $x \in \mathbb{R}^n$  ( $k = 0, 1, \dots, m$ ). Let  $f_0^*$  be the optimal value of the POP (18.1);  $f_0^*$  may be  $-\infty$  if the POP (18.1) is unbounded and  $+\infty$  if it is infeasible.

A hierarchy of semidefinite programming (SDP) relaxations proposed by Lasserre in [22] is known as a powerful method for computing a global optimal solution of the POP (18.1). The hierarchy is arranged according to a positive integer, called the relaxation order in this chapter. It determines qualities and sizes of SDP relaxation problems in the hierarchy. Each SDP problem with

---

S. Kim (✉)

Department of Mathematics, Ewha W. University, 11-1 DaHyun-Dong,  
SuhDaeMoon-Gu, Seoul, Korea 120-750  
e-mail: [skim@ewha.ac.kr](mailto:skim@ewha.ac.kr)

M. Kojima

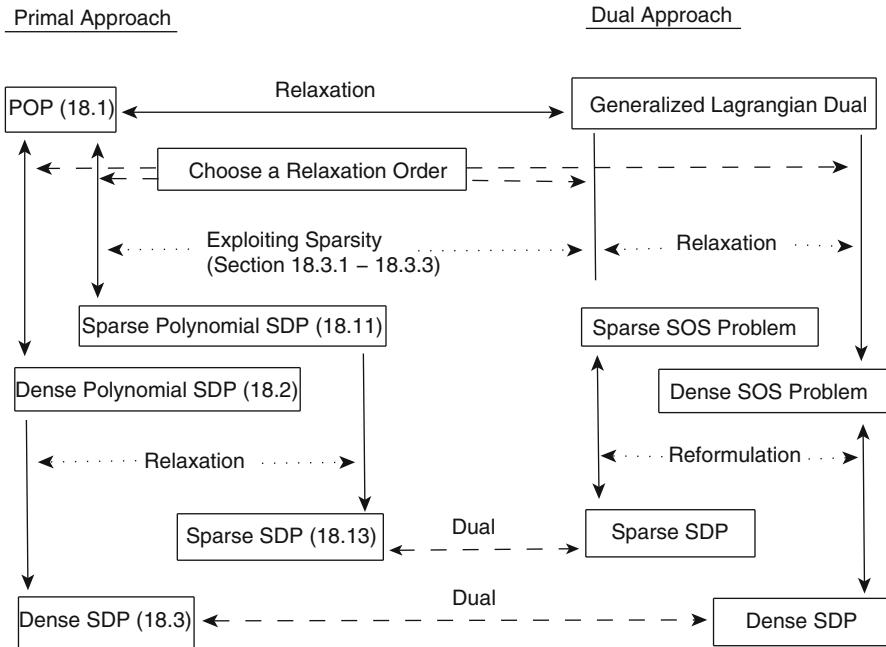
Department of Mathematical and Computing Sciences, Tokyo Institute of Technology,  
2-12-1-W8-29, Oh-Okayama, Meguro, Tokyo 152-8552, Japan  
e-mail: [kojima@is.titech.ac.jp](mailto:kojima@is.titech.ac.jp)

a relaxation order  $\omega$  provides a lower bound  $\zeta_\omega$  for the optimal objective value  $f_0^*$  of the POP (18.1). It was established in [22] that  $\zeta_\omega$  converges  $f_0^*$  as  $\omega \rightarrow \infty$  under a moderate assumption that requires the compactness of the feasible region of (18.1). The size of the SDP relaxation, however, increases very rapidly as the number of variables of (18.1), the degree of polynomials involved in (18.1), and/or the relaxation order  $\omega$  increase. In practice, it is often difficult to obtain an approximation to the global optimal solution of (18.1) because the resulting SDP relaxation is too large to solve. Without employing techniques to reduce the size of SDP relaxations, an approximation to the global optimal solution of a medium-to-large-scale POP is difficult to obtain. One important technique to cope with this difficulty is exploiting structured sparsity of POPs, which is the subject of this chapter. In many cases of POPs with a large number of variables, every  $f_k$  ( $k = 1, 2, \dots, m$ ) involves a small subset of variables, and the objective function  $f_0$  is a sum of polynomials, each of which involves a small subset of variables. The sparsity described in this chapter is concerned with a structured interaction among the variables over the polynomials, which is characterized in terms of a chordal graph.

The purpose of this chapter is to present a survey of the sparse SDP relaxation, which was originally proposed in [34] as a sparse variant of Lasserre's SDP relaxation [22]. We call Lasserre's SDP relaxation the dense SDP relaxation. The focus is on the algorithmic aspects of the sparse SDP relaxation. For its theoretical convergence, we refer to [23]. Figure 18.1 shows an overview of the dense and sparse SDP relaxations. We can describe these SDP relaxations in two methods: A primal approach, which is our primary concern in this chapter, and a dual approach.

In the primal approach, we first choose a relaxation order  $\omega$ , and then, convert the POP (18.1) into an equivalent polynomial SDP (PSDP) by adding valid polynomial matrix inequalities. The relaxation order  $\omega$  is used at this stage to restrict the degree of the valid polynomial matrix inequalities added to at most  $2\omega$ . Thus, it controls the size and quality of the SDP relaxation that will be derived. At this stage, we can also incorporate the sparsity characterized by a chordal graph structure to construct a sparse PSDP. After expanding real and matrix-valued polynomial functions in the sparse (or dense) PSDP, the linearization by replacing each monomial by a single real variable is followed to obtain the sparse (or dense) SDP relaxation.

In the dual approach, a generalized Lagrangian relaxation to the POP (18.1) is applied to obtain a generalized Lagrangian dual [14, 22, 28] of (1) that includes sums of squares (SOS) polynomials for Lagrangian multipliers. After choosing a relaxation order  $\omega$ , we perform SOS relaxation for the Lagrangian dual. The relaxation order  $\omega$  is chosen to restrict the degrees of SOS polynomials used there by at most  $2\omega$ . As a result, it controls the quality and the size of the SOS relaxation. As in the primal approach, the same structured sparsity can be exploited at this stage to obtain a sparse SOS problem. Finally, the sparse (or dense) SOS relaxation



**Fig. 18.1** Overview of Lasserre’s hierarchy of (dense) SDP relaxations [22] and its sparse variant [34]

problem is reformulated as a sparse (or dense) SDP. We note that the sparse (or dense) SDP obtained in the primal and dual approaches have a primal-dual relationship. In this chapter, we restrict our discussion to the primal approach, and we refer to [22, 34] for readers who are interested in the dual approach.

In Sect. 18.2, we describe Lasserre’s dense SDP relaxation of the POP (18.1) after introducing notation and symbols used throughout this chapter. Section 18.3 is devoted to the primal approach to (18.1). In Sect. 18.3.1, we present a class of SDP problems having multiple but small-sized matrix variables that can be solved efficiently by primal-dual interior-point methods [3, 29, 32, 33, 36]. This class of SDPs serves as target SDPs into which POPs are aimed to be relaxed. In Sect. 18.3.2, a sparse Cholesky factorization and a chordal graph are introduced. These are used in Sect. 18.3.3 for formulating structured sparsity from POPs. The sparse SDP relaxation of the POP (18.1) is described in Sect. 18.3.4. Section 18.4 contains additional techniques used in the software package SparsePOP [31, 35], which is an implementation of the sparse SDP relaxation. Numerical results on SparsePOP are reported in Sect. 18.5, and the applications of the sparse SDP relaxation to the sensor network localization [15, 16, 30] are presented in Sect. 18.6.

## 18.2 Preliminaries

### 18.2.1 Notation and Symbols

Throughout this chapter we let  $\mathbb{R}$  be the set of real numbers, and  $\mathbb{Z}_+$  the set of nonnegative integers. We use  $\mathbb{R}^n$  for the  $n$ -dimensional Euclidean space, and  $\mathbb{Z}_+^n$  for the set of nonnegative integer vectors in  $\mathbb{R}^n$ . Each element  $x$  of  $\mathbb{R}^n$  is an  $n$ -dimensional column vector of  $x_i \in \mathbb{R}$  ( $i = 1, 2, \dots, n$ ), written as  $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$ , and  $x^T$  denotes the  $n$ -dimensional row vector. Let  $\mathbb{S}^n$  denote the space of  $n \times n$  real symmetric matrices and  $\mathbb{S}_+^n \subset \mathbb{S}^n$  the cone of positive semidefinite matrices. We write  $Y \succeq O$  if  $Y \in \mathbb{S}_+^n$  for some  $n$ .

$\mathbb{R}[x]$  is the set of real-valued multivariate polynomials in  $x_i$  ( $i = 1, 2, \dots, n$ ). Each  $f \in \mathbb{R}[x]$  is expressed as  $f(x) = \sum_{\alpha \in \mathcal{F}} c(\alpha)x^\alpha$  for some nonempty finite subset  $\mathcal{F}$  of  $\mathbb{Z}_+^n$  and  $c(\alpha) \in \mathbb{R}$  ( $\alpha \in \mathcal{F}$ ), where  $x^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$  for every  $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$  and every  $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_n) \in \mathbb{Z}_+^n$ . We define the support of  $f$  by  $\text{supp}(f) = \{\alpha \in \mathcal{F} : c(\alpha) \neq 0\}$  and the degree of  $f$  by  $\deg(f) = \max\{\sum_{i=1}^n \alpha_i : \alpha \in \text{supp}(f)\}$ .

For every nonempty finite subset  $\mathcal{G}$  of  $\mathbb{Z}_+^n$ , let  $\mathbb{R}^{\mathcal{G}}$  denote the  $|\mathcal{G}|$ -dimensional Euclidean space whose coordinates are indexed by  $\alpha \in \mathcal{G}$ . Each vector of  $\mathbb{R}^{\mathcal{G}}$  is denoted as a column vector  $w = (w_\alpha : \alpha \in \mathcal{G})$ . The indices  $\alpha \in \mathcal{G}$  can be ordered arbitrary except for the element  $\mathbf{0} \in \mathbb{Z}_+^n$ , which is assumed to be the first index whenever  $\mathbf{0} \in \mathcal{G}$ . We use the symbol  $\mathbb{S}^{\mathcal{G}}$  for the set of  $|\mathcal{G}| \times |\mathcal{G}|$  symmetric matrices with coordinates  $\alpha \in \mathcal{G}$ . Let  $\mathbb{S}_+^{\mathcal{G}}$  be the set of positive semidefinite matrices in  $\mathbb{S}^{\mathcal{G}}$ ;  $V \in \mathbb{S}_+^{\mathcal{G}}$  iff  $w^T V w = \sum_{\alpha \in \mathcal{G}} \sum_{\beta \in \mathcal{G}} V_{\alpha\beta} w_\alpha w_\beta \geq 0$  for every  $w = (w_\alpha : \alpha \in \mathcal{G}) \in \mathbb{R}^{\mathcal{G}}$ .

For every nonempty finite subset  $\mathcal{G}$  of  $\mathbb{Z}_+^n$ , let  $u(x, \mathcal{G})$  denote the  $|\mathcal{G}|$ -dimensional column vector of elements  $x^\alpha$  ( $\alpha \in \mathcal{G}$ );  $u(x, \mathcal{G}) = (x^\alpha : \alpha \in \mathcal{G}) \in \mathbb{R}^{\mathcal{G}}$  for every  $x \in \mathbb{R}^n$ . Obviously,  $u(x, \mathcal{G})u(x, \mathcal{G})^T \in \mathbb{S}_+^{\mathcal{G}}$  for every  $x \in \mathbb{R}^n$ . For every  $\psi \in \mathbb{Z}_+$ , let  $\mathcal{A}_\psi = \{\alpha \in \mathbb{Z}_+^n : \sum_{i=1}^n \alpha_i \leq \psi\}$ . Then, we see that  $\mathbf{0} \in \mathcal{A}_\psi$ . For any  $x \in \mathbb{R}^n$ , the upper left element of the matrix  $u(x, \mathcal{A}_\psi)u(x, \mathcal{A}_\psi)^T \in \mathbb{S}_+^{\mathcal{A}_\psi}$  is the constant  $x^0 = 1$ . We also note that  $\mathcal{A}_\psi + \mathcal{A}_\psi = \mathcal{A}_{2\psi}$  for every  $\psi \in \mathbb{Z}_+$ , where  $\mathcal{G} + \mathcal{H}$  denotes the Minkowski sum of two  $\mathcal{G}, \mathcal{H} \subset \mathbb{Z}_+^n$ , i.e.,  $\mathcal{G} + \mathcal{H} = \{\alpha + \beta : \alpha \in \mathcal{G}, \beta \in \mathcal{H}\}$ . These facts are used in the next subsection.

We represent each polynomial  $f_k$  in the POP (18.1) as  $f_k(x) = \sum_{\alpha \in \mathcal{F}_k} c_k(\alpha)x^\alpha$  for every  $x \in \mathbb{R}^n$  ( $k = 0, 1, \dots, m$ ). We may assume without loss of generality that  $\mathbf{0} \notin \mathcal{F}_0$ . We also let  $\omega_k = \lceil \deg(f_k)/2 \rceil$  ( $k = 0, 1, \dots, m$ ) and  $\omega_{\max} = \max\{\omega_k : k = 0, 1, \dots, m\}$ . We use the following examples to illustrate the dense and sparse SDP relaxations in the subsequent discussions.

*Example 18.1.*

$$\begin{aligned} & \text{minimize} && f_0(x) = x_2 - 2x_1x_2 + x_2x_3 \\ & \text{subject to} && f_1(x) = 1 - x_1^2 - x_2^2 \geq 0, \quad f_2(x) = 1 - x_2^2 - x_3^2 \geq 0. \end{aligned}$$

Notice that  $\omega_0 = \lceil \deg(f_0)/2 \rceil = 1, \omega_1 = \lceil \deg(f_1)/2 \rceil = 1, \omega_2 = \lceil \deg(f_2)/2 \rceil = 1$  and  $\omega_{\max} = 1$ .

*Example 18.2.*

$$\begin{aligned} \text{minimize } & f_0(x) = \sum_{i=1}^n (a_i x_i^\gamma + b_i x_i^{\gamma-1}) + cx_1 x_n \\ \text{subject to } & f_k(x) = 1 - x_k^2 - x_{k+1}^2 \geq 0 \quad (k = 1, 2, \dots, n-1), \end{aligned}$$

where  $a_i$  ( $i = 1, 2, \dots, n$ ),  $b_i$  ( $i = 1, 2, \dots, n$ ) and  $c$  are real constants chosen randomly from  $[-1, 1]$  and  $\gamma$  is a positive integer not less than 2. We see that  $\omega_0 = \lceil \deg(f_0)/2 \rceil = \lceil \gamma/2 \rceil$ ,  $\omega_k = \lceil \deg(f_k)/2 \rceil = 1$  ( $k = 1, 2, \dots, n-1$ ) and  $\omega_{\max} = \lceil \gamma/2 \rceil$ .

### 18.2.2 Lasserre's Dense SDP Relaxation of a POP

Lasserre's SDP relaxation method [22] for a POP generates a hierarchy of SDP problems, parametrized by an integer parameter  $\omega \geq \omega_{\max}$ . Solving the hierarchy of SDP problems provides a sequence of monotonically nondecreasing lower bounds  $\{\zeta_\omega^d : \omega \geq \omega_{\max}\}$  for the optimal value of the POP. We call each problem in the hierarchy the dense SDP relaxation problem with the *relaxation order*  $\omega$  in this chapter. After choosing a relaxation order  $\omega \geq \omega_{\max}$ , we first transform the POP (18.1) into an equivalent polynomial PSDP (PSDP)

$$\left. \begin{aligned} \text{minimize } & f_0(x) \\ \text{subject to } & u(x, \mathcal{A}_{\omega-\omega_k}) u(x, \mathcal{A}_{\omega-\omega_k})^T f_k(x) \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}} \\ & (k = 1, 2, \dots, m), \\ & u(x, \mathcal{A}_\omega) u(x, \mathcal{A}_\omega)^T \in \mathbb{S}_+^{\mathcal{A}_\omega}. \end{aligned} \right\} \quad (18.2)$$

To verify the equivalence between the POP (18.1) and the PSDP (18.2), we first observe that they have the same objective function. If  $x \in \mathbb{R}^n$  is a feasible solution of the POP (18.1), then it is a feasible solution of the PSDP (18.2) because the symmetric matrices  $u(x, \mathcal{A}_{\omega-\omega_k}) u(x, \mathcal{A}_{\omega-\omega_k})^T$  and  $u(x, \mathcal{A}_\omega) u(x, \mathcal{A}_\omega)^T$  are positive semidefinite for any  $x \in \mathbb{R}^n$ . The converse is also true because the symmetric matrices  $u(x, \mathcal{A}_{\omega-\omega_k}) u(x, \mathcal{A}_{\omega-\omega_k})^T$  ( $k = 1, 2, \dots, m$ ) have the element  $x^0 = 1$  in their upper-left corner. This confirms the equivalence.

Let  $\mathcal{F}^d = (\mathcal{A}_\omega + \mathcal{A}_\omega) = \mathcal{A}_{2\omega}$  denote the set of all monomials involved in the PSDP (18.2). Since the objective function is a real-valued polynomial and the left-hand side of the matrix inequality constraints are real symmetric matrix-valued polynomials, we can rewrite the PSDP (18.2) as

$$\text{minimize} \quad \sum_{\alpha \in \mathcal{F}^d} c_0^d(\alpha) x^\alpha$$

$$\begin{aligned} \text{subject to } & \sum_{\alpha \in \mathcal{F}^d} L_k^d(\alpha, \omega) x^\alpha \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}} \quad (k = 1, \dots, m), \\ & \sum_{\alpha \in \mathcal{F}^d} M^d(\alpha, \omega) x^\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega}, \end{aligned}$$

for some  $c_0^d(\alpha) \in \mathbb{R}$  ( $\alpha \in \mathcal{F}^d$ ), real symmetric matrices  $L_k^d(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^d$ ,  $k = 1, \dots, m$ ) and  $M^d(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^d$ ). Replacing each monomial  $x^\alpha$  by a single variable  $y_\alpha \in \mathbb{R}$  provides the dense SDP relaxation problem of the POP (18.1):

$$\left. \begin{aligned} \text{minimize } & \sum_{\alpha \in \mathcal{F}^d} c_0^d(\alpha) y_\alpha \\ \text{subject to } & \sum_{\alpha \in \mathcal{F}^d} L_k^d(\alpha, \omega) y_\alpha \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}} \quad (k = 1, \dots, m), \\ & \sum_{\alpha \in \mathcal{F}^d} M^d(\alpha, \omega) y_\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega}, \quad y_0 = 1. \end{aligned} \right\} \quad (18.3)$$

(Note that  $\mathbf{0} \in \mathcal{F}^d = \mathcal{A}_{2\omega}$  and  $x^0 = 1$ ). If  $x \in \mathbb{R}^n$  is a feasible solution of the PSDP (18.3),  $(y_\alpha : \alpha \in \mathcal{F}^d) = (x^\alpha : \alpha \in \mathcal{F}^d)$  is a feasible solution of (18.3) with the same objective value  $\sum_{\alpha \in \mathcal{F}^d} c_0^d(\alpha) y_\alpha$  as the objective value  $\sum_{\alpha \in \mathcal{F}^d} c_0^d(\alpha) x^\alpha$  of the PSDP. This implies that (18.3) is a relaxation of the PSDP, hence, a relaxation of (18.1).

Using Example 18.1, we illustrate the dense SDP relaxation. If we take the relaxation order  $\omega = \omega_{\max} = 1$ , then the PSDP (18.2) reads

$$\begin{aligned} \text{minimize } & x_2 - 2x_1x_2 + x_2x_3 \\ \text{subject to } & 1^2 \cdot (1 - x_1^2 - x_2^2) \geq 0, \quad 1^2 \cdot (1 - x_2^2 - x_3^2) \geq 0, \\ & \begin{pmatrix} x^{000} & x^{100} & x^{010} & x^{001} \\ x^{100} & x^{200} & x^{110} & x^{101} \\ x^{010} & x^{110} & x^{020} & x^{011} \\ x^{001} & x^{101} & x^{011} & x^{002} \end{pmatrix} = \begin{pmatrix} 1 & x_1 & x_2 & x_3 \\ x_1 & x_1^2 & x_1x_2 & x_1x_3 \\ x_2 & x_1x_2 & x_2^2 & x_2x_3 \\ x_3 & x_1x_3 & x_2x_3 & x_3^2 \end{pmatrix} \in \mathbb{S}_+^{\mathcal{A}_\omega}. \end{aligned}$$

Here we simply write  $x^{\alpha_1\alpha_2\alpha_3}$  instead of  $x^{(\alpha_1, \alpha_2, \alpha_3)}$ . Replacing each  $x^\alpha$  by  $y_\alpha$ , we obtain an SDP relaxation problem:

$$\begin{aligned} \text{minimize } & y_{010} - 2y_{110} + y_{011} \\ \text{subject to } & y_{000} - y_{200} - y_{020} \geq 0, \quad y_{000} - y_{020} - y_{002} \geq 0, \\ & \begin{pmatrix} y_{000} & y_{100} & y_{010} & y_{001} \\ y_{100} & y_{200} & y_{110} & y_{101} \\ y_{010} & y_{110} & y_{020} & y_{011} \\ y_{001} & y_{101} & y_{011} & y_{002} \end{pmatrix} \in \mathbb{S}_+^{\mathcal{A}_\omega}, \quad y_{000} = 1. \end{aligned}$$

Let

$$\begin{aligned}\mathcal{F}^d = \mathcal{A}_{2\omega} = & \{(0,0,0), (1,0,0), (0,1,0), (0,0,1), (2,0,0), \\ & (1,1,0), (1,0,1), (0,2,0), (0,1,1), (0,0,2)\}.\end{aligned}$$

Then, we can rewrite the previous SDP relaxation problem as

$$\left. \begin{array}{l} \text{minimize } y_{010} - 2y_{110} + y_{011} \\ \text{subject to } y_{000} - y_{200} - y_{020} \geq 0, y_{000} - y_{020} - y_{002} \geq 0, \\ \sum_{\alpha \in \mathcal{F}^d} M^d(\alpha, \omega) y_\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega}, y_{000} = 1, \end{array} \right\} \quad (18.4)$$

for some  $M^d(\alpha, \omega) \in \mathbb{S}^{\mathcal{A}_\omega}$  ( $\alpha \in \mathcal{F}^d$ ); for example,

$$M^d((1,0,0), \omega) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad M^d((0,1,1), \omega) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Let  $\zeta_\omega^d$  denote the optimal value of (18.3). Then,  $\zeta_\omega^d \leq \zeta_{\omega+1}^d \leq f_0^*$  for every  $\omega \geq \omega_{\max}$ , where  $f_0^*$  denotes the optimal value of the POP (18.1). Under a moderate assumption that requires the compactness of the feasible region of (18.1),  $\zeta_\omega^d$  converges  $f_0^*$  as  $\omega \rightarrow \infty$  [22].

We note that

$$\text{the size of } L_k^d(\alpha, \omega) = |\mathcal{A}_{\omega-\omega_k}| = \binom{n+\omega-\omega_k}{\omega-\omega_k} \quad (k = 1, 2, \dots, n-1),$$

$$\text{the size of } M^d(\alpha, \omega) = |\mathcal{A}_\omega| = \binom{n+\omega}{\omega},$$

$$\text{the number of variables } = |\mathcal{F}^d| = \binom{n+2\omega}{2\omega},$$

which increase rapidly with  $n$  and/or  $\omega$ . Therefore, solving the dense SDP relaxation problem (18.3) becomes increasingly time-consuming and difficult as  $n$  and/or  $\omega$  grow.

Numerical results are presented in Table 18.1 to show the growth of the three numbers of the dense SDP relaxation applied to Example 18.2. We needed to take  $\omega \geq \omega_{\max} = \lceil \gamma/2 \rceil$ . In Table 18.1, notice that the size of  $L_k^d(\alpha, \omega)$ , the size of  $M^d(\alpha, \omega)$  and the number of variables  $|\mathcal{F}^d|$  increase rapidly as  $n$  and/or  $\omega$  becomes large. Large values of  $n$  and  $\omega$  often resulted in out-of-memory error, as a result, SDPA [6, 29, 36] could not solve the SDP relaxation problem. We were able to obtain an accurate

**Table 18.1** Numerical results on the dense SDP relaxation applied to Example 18.2. eTime denotes the elapsed time to solve the SDP problem by the Matlab version of SDPA 7.3.1 [29] on 3.06 GHz Intel Core 2 Duo with 8HB memory

$\gamma$	$n$	$\omega = \omega_{\max}$	Size $L_k^d$	Size $M^d$	$ \mathcal{F}^d $	eTime
2	10	1	1	11	66	0.05
2	20	1	1	21	231	0.08
2	40	1	1	41	861	1.08
4	10	2	11	66	1,001	2.23
4	20	2	21	231	10,626	1063.16
4	40	2	41	861	135,751	Out-of-memory
6	10	3	66	286	8,008	721.21
6	20	3	231	1771	230,230	Out-of-memory
6	40	3	861	12,341	9,366,819	Out-of-memory

approximation to the optimal solution when SDPA could solve its SDP relaxation problem.

### 18.3 Sparse SDP Relaxations of a POP

As shown in Sect. 18.2, POPs are transformed and relaxed into SDPs. We first present a strategy to exploit sparsity in a general SDP that will be solved by the primal-dual interior-point method.

#### 18.3.1 Semidefinite Programming Problems That Can Be Efficiently Solved by the Primal-Dual Interior-Point Method

The class of SDPs described in this section are the target SDPs into which POPs with sparsity are aimed to be relaxed for computational efficiency.

Consider the equality standard form of SDP and its dual

$$\begin{aligned} & \text{minimize } A_0 \bullet X \\ & \text{subject to } A_p \bullet X = b_p \quad (p = 1, 2, \dots, m), \quad X \succeq O, \end{aligned} \quad \left. \right\} \quad (18.5)$$

$$\begin{aligned} & \text{maximize } \sum_{p=1}^m b_p y_p \\ & \text{subject to } A_0 - \sum_{p=1}^m A_p y_p \succeq O, \end{aligned} \quad (18.6)$$

where  $A_p \in \mathbb{S}^n$  ( $p = 0, 1, \dots, m$ ). We call the SDP (18.6) the linear matrix inequality (LMI) form of SDP. If no assumption is made on the sparsity of the coefficient matrices  $A_p \in \mathbb{S}^n$  ( $p = 0, 1, \dots, m$ ), the efficiency of solving the SDP by the primal-dual interior-point method [3, 32, 33, 36] mainly depends on two factors. The first is the size  $n$  of the matrix variable  $X$  in (18.5) (or the size of the LMI constraint in (18.6)), and the second is the number  $m$  of equalities in (18.5) (or the number of the real variables  $y_p$  ( $p = 1, 2, \dots, m$ )). We note that the number  $m$  determines the size of the Schur complement matrix, the  $m \times m$  positive definite coefficient matrix of the Schur complement equation, which is solved at each iteration of the primal-dual interior-point method. If either  $n$  or  $m$  increases, more elapsed time and memory are required to solve the SDP. See [6], for example. Increasing the efficiency of the primal-dual interior-point method can be achieved by exploiting sparsity. We discuss this issue in detail with the LMI form of SDP (18.6).

The simplest structured sparsity that makes the primal-dual interior-point method work efficiently is a block-diagonal structure of the coefficient matrices. Suppose that each  $A_p$  is of the form

$$A_p = \text{diag}(A_{p1}, A_{p2}, \dots, A_{p\ell}) = \begin{pmatrix} A_{p1} & O & \cdots & O \\ O & A_{p2} & \cdots & O \\ \vdots & \vdots & \ddots & \vdots \\ O & O & \cdots & A_{p\ell} \end{pmatrix} \quad (18.7)$$

( $p = 0, 1, \dots, m$ ), where  $A_{pj} \in \mathbb{S}^{n_j}$  ( $j = 1, 2, \dots, \ell$ ,  $p = 0, 1, \dots, m$ ). Note that for each  $j$ , the matrices  $A_{0j}, A_{1j}, \dots, A_{mj}$  are of a same size  $n_j \times n_j$ , and that  $\sum_{j=1}^{\ell} n_j = n$ . In this case, we can rewrite the LMI form of SDP (18.6) as

$$\left. \begin{array}{l} \text{maximize} \quad \sum_{p=1}^m b_p y_p \\ \text{subject to} \quad A_{0j} - \sum_{p=1}^m A_{pj} y_p \succeq O \quad (j = 1, 2, \dots, \ell). \end{array} \right\} \quad (18.8)$$

If we compare the original LMI form of SDP (18.6) with the transformed SDP (18.8), we notice that the single  $n \times n$  matrix inequality is decomposed into multiple matrix inequalities of reduced size. Thus, if their sizes are small, the SDP (18.8) can be solved more efficiently than the SDP (18.6).

Notice that the SDP (18.8) involves the same number of real variables  $y_p$  ( $p = 1, 2, \dots, m$ ) as in the original SDP (18.6), which indicates that the size of the Schur complement equation remains the same. Nevertheless, the SDP (18.8) can have a considerable advantage over the SDP (18.6). To see this clearly, let us assume without loss of generality that all coefficient matrix  $A_p$  ( $p = 1, 2, \dots, m$ ) are nonzero in (18.6). In fact, if some  $A_p = O$ , then we must have  $b_p = 0$  since otherwise the SDP (18.6) is unbounded. On the other hand, some of the block matrices  $A_{p1}, A_{p2}, \dots, A_{p\ell}$  can be zero in (18.7), although the entire matrix  $A_p$  is not,

for each  $p = 1, 2, \dots, m$ . Then, the Schur complement matrix often becomes sparse so that the sparse Cholesky factorization can be applied to the Schur complement matrix.

In the primal-dual interior-point method [3, 32, 33, 36], each element  $B_{pq}$  of the  $m \times m$  Schur complement matrix  $B$  is given by  $B_{pq} = \sum_{j=1}^{\ell} T_j A_{pj} U_j \bullet A_{qj}$  ( $p, q = 1, 2, \dots, m$ ) for fully dense  $T_j \in \mathbb{S}^{n_j}$ ,  $U_j \in \mathbb{S}^{n_j}$  ( $j = 1, 2, \dots, \ell$ ). For details, we refer to [6]. Notice that  $B_{pq}$  is nonzero if and only if there is a  $j$  such that both the coefficient matrix  $A_{pj}$  of the variable  $y_p$  and the coefficient matrix  $A_{qj}$  of the variable  $y_q$  are nonzero matrices. Thus, if we define the  $m \times m$  symbolic matrix  $R$  by

$$R_{pq} = \begin{cases} \star & \text{if } A_{pj} \neq O \text{ and } A_{qj} \neq O \text{ for some } j = 1, 2, \dots, \ell, \\ 0 & \text{otherwise.} \end{cases}$$

$R$  represents the sparsity pattern of the Schur complement matrix  $B$ . The matrix  $R$  is called *the correlative sparsity pattern matrix (abbreviated as the csp matrix)* of the SDP (18.8). A class of SDPs of the form (18.8) that can be solved efficiently by the primal-dual interior-point method is characterized by the conditions:

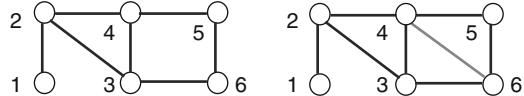
- (A) The sizes of the coefficient matrices  $A_{pj}$  ( $j = 1, 2, \dots, \ell$ ,  $p = 1, 2, \dots, m$ ) are small.
- (B) The csp matrix  $R$  allows a (symbolic) sparse Cholesky factorization.

Let us consider the dense SDP relaxation problem for these conditions. The dense SDP relaxation problem (18.3) of the POP (18.1) is of the form (18.8). As discussed at the end of Sect. 18.2.2, the size of each coefficient matrix increases rapidly as  $n$  or  $\omega$  becomes large. See Table 18.1. Furthermore, its csp matrix is fully dense, resulting in the dense Schur complement matrix. This was a main reason for “out-of-memory” error in Table 18.1. Thus, the dense SDP relaxation problem (18.3) satisfies neither of the conditions (A) and (B).

### 18.3.2 A Sparse Cholesky Factorization and a Chordal Graph

Let  $R$  be a symbolic symmetric matrix that represents the sparsity pattern of a class of  $n \times n$  symmetric matrices introduced in the previous subsection. We assume that the nonzero symbol  $\star$  is assigned to all diagonal elements and some off-diagonal elements. It is well-known that a sparse Cholesky factorization is characterized in terms of a chordal graph. We call an undirected graph *chordal* if every cycle of length  $\geq 4$  has a chord, i.e., an edge joining two nonconsecutive vertices of the cycle. For  $R$ , an undirected graph  $G(N, E)$  is defined with the node set  $N = \{1, 2, \dots, n\}$  and the edge set  $E$  such that  $(i, j) \in E$  if and only if  $R_{ij} = \star$  and  $i > j$ . Note that edge  $(j, i)$  is identified with  $(i, j)$ . The graph  $G(N, E)$  is called *the sparsity pattern graph* of  $R$ . If the graph  $G(N, E)$  is chordal, then there exists a permutation matrix  $P$  such that the matrix  $PRP^T$  can be factorized (symbolically) as  $PRP^T = LL^T$  with no fill-in, where  $L$  denotes a lower triangular matrix. The matrix  $P$  is obtained from a perfect

**Fig. 18.2**  $G(N, E)$  on the left vs.  $G(N, \bar{E})$  on the right



elimination ordering. The maximal cliques of a chordal graph are computed with reference to the perfect elimination ordering. In fact, let  $C_k = \{i : [P^T L]_{ik} = \star\} \subset N$  ( $k = 1, 2, \dots, n$ ). Then, the maximal sets of the family of sets  $C_k$  ( $k = 1, 2, \dots, n$ ) form the maximal cliques of the chordal graph  $G(N, E)$ .

If the graph  $G(N, E)$  is not chordal, a chordal extension of  $G(N, E)$  can be constructed using the sparse Cholesky factorization. Define the  $n \times n$  symbolic sparsity pattern matrix  $R$  by

$$R_{pq} = \begin{cases} \star & \text{if } p = q, (p, q) \in E \text{ or } (q, p) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

A simultaneous row and column permutation to  $R$  such as the symmetric minimum degree ordering can be applied before the factorization of  $R$ . Let  $P$  be the permutation matrix corresponding to such a permutation. Applying the Cholesky factorization to  $PRP^T$ , we obtain a lower triangular matrix  $L$  such that  $PRP^T = LL^T$ . Then the edge set  $\bar{E}$  of a chordal extension  $G(N, \bar{E})$  of  $G(N, E)$  is obtained by  $\bar{E} = \{(i, j) : i \neq j, [P^T L]_{ij} = \star\}$ , and its maximal cliques can be chosen from the family of cliques  $C_k = \{i : [P^T L]_{ik} = \star\} \subset N$  ( $k = 1, 2, \dots, n$ ) as described previously. For the basic definition and properties of chordal graphs, we refer to [9].

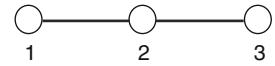
*Example 18.3.* Consider the sparsity pattern matrix

$$R = \begin{pmatrix} \star & \star & 0 & 0 & 0 & 0 \\ \star & \star & \star & \star & 0 & 0 \\ 0 & \star & \star & \star & 0 & \star \\ 0 & \star & \star & \star & \star & 0 \\ 0 & 0 & 0 & \star & \star & \star \\ 0 & 0 & \star & 0 & \star & \star \end{pmatrix},$$

which yields the sparsity pattern graph  $G(N, E)$  on the left of Fig. 18.2. This graph is not chordal because the cycle consisting of 4 edges  $(3, 4)$ ,  $(4, 5)$ ,  $(5, 6)$ ,  $(6, 3)$  does not have a chord. A chordal extension  $G(N, \bar{E})$  of the graph  $G(N, E)$  is shown on the right of Fig. 18.2. In this case, if the identity matrix is chosen for  $P$  and the Cholesky factorization is applied to  $R$ , the lower triangular matrix satisfying  $R = LL^T$  is

$$L = \begin{pmatrix} \star & 0 & 0 & 0 & 0 & 0 \\ \star & \star & 0 & 0 & 0 & 0 \\ 0 & \star & \star & 0 & 0 & 0 \\ 0 & \star & \star & \star & 0 & 0 \\ 0 & 0 & 0 & \star & \star & 0 \\ 0 & 0 & \star & \star & \star & \star \end{pmatrix}.$$

**Fig. 18.3** The csp graph  $G(N, E)$  of the POP in Example 18.1



Note that a fill-in occurs in the (6,4)th element, which corresponds to the edge (6,4) of the chordal extension  $G(N, \bar{E})$ . Each column of  $L$  leads a clique, thus, the resulting 6 cliques are  $\{1,2\}$ ,  $\{2,3,4\}$ ,  $\{3,4,6\}$ ,  $\{4,5,6\}$ ,  $\{5,6\}$  and  $\{6\}$ . Choosing the maximal ones from them, we have the 4 maximal cliques

$$\{1,2\}, \{2,3,4\}, \{3,4,6\} \text{ and } \{4,5,6\} \quad (18.9)$$

of the chordal extension  $G(N, \bar{E})$ .

### 18.3.3 Formulating Structured Sparsity

The sparsity that can be extracted from the POP (18.1) for a sparse SDP relaxation is discussed in this subsection. We call this sparsity *the correlative sparsity*. Let  $N = \{1, 2, \dots, n\}$ . For every  $k = 1, 2, \dots, m$ , the set of indices  $i$  of variables  $x_i$  in the  $k$ th inequality  $f_k(x) \geq 0$  is defined:  $F_k = \{i \in N : c_k(\alpha) \neq 0 \text{ and } \alpha_i \geq 1 \text{ for some } \alpha \in \mathcal{F}_k\}$ . We construct an undirected graph  $G(N, E)$  to represent the sparsity structure of (18.1) by connecting a pair  $(i, j)$  with  $i \neq j$  selected from the node set  $N$  as an edge, i.e.,  $(i, j) \in E$  if and only if either there is an  $\alpha \in \mathcal{F}_0$  such that  $c_0(\alpha) \neq 0$ ,  $\alpha_i > 0$  and  $\alpha_j > 0$  or  $i, j \in F_k$  for some  $k = 1, 2, \dots, m$ . We identify each edge  $(i, j)$  with  $i > j$  with the edge  $(j, i)$ . The graph  $G(N, E)$  constructed this way is called *the correlative sparsity pattern (csp) graph*. Let  $G(N, \bar{E})$  be a chordal extension of  $G(N, E)$ , and  $C_1, C_2, \dots, C_\ell$  be its maximal cliques. Since each  $F_k$  is a clique of  $G(N, E)$ , we can take a maximal clique  $\tilde{C}_k \in \{C_1, C_2, \dots, C_\ell\}$  such that

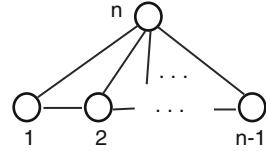
$$F_k \subset \tilde{C}_k \quad (k = 1, 2, \dots, m). \quad (18.10)$$

For Example 18.1,  $N = \{1, 2, 3\}$ ,  $F_1 = \{1, 2\}$ ,  $F_2 = \{2, 3\}$  and  $E = \{(1, 2), (2, 3)\}$ . Figure 18.3 shows the csp graph  $G(N, E)$ , which is apparently chordal because there is no cycle, and the maximal cliques are  $C_1 = \{1, 2\}$  and  $C_2 = \{2, 3\}$ . Hence, we can take  $\tilde{C}_1 = \{1, 2\} = F_1$  and  $\tilde{C}_2 = \{2, 3\} = F_2$ .

*Example 18.4.*

$$\begin{aligned} & \text{minimize} \quad - \sum_{i=1}^6 x_i^2 \\ & \text{subject to} \quad x_1^2 + x_2^2 \leq 1, \quad x_2 + x_3^2 + x_4^2 \leq 1, \\ & \quad x_4 + x_5^2 \leq 1, \quad x_3 + x_6^2 \leq 1, \quad x_5^2 + x_6^2 \leq 1. \end{aligned}$$

**Fig. 18.4** A chordal extension  $G(N, \bar{E})$  of the csp graph  $G(N, E)$  of the POP in Example 18.2



In this case, we have  $F_1 = \{1, 2\}$ ,  $F_2 = \{2, 3, 4\}$ ,  $F_3 = \{4, 5\}$ ,  $F_4 = \{3, 6\}$  and  $F_5 = \{5, 6\}$ . We also see that the csp graph  $G(N, E)$  coincides with the one on the left of Fig. 18.2. Thus, the graph on the right of Fig. 18.2 is a chordal extension of the csp graph  $G(N, E)$ , and its maximal cliques are given in (18.9). Consequently, we can take  $\tilde{C}_k$  ( $k = 1, 2, \dots, 5$ ) as  $\tilde{C}_1 = \{1, 2\}$ ,  $\tilde{C}_2 = \{2, 3, 4\}$ ,  $\tilde{C}_3 = \{4, 5, 6\}$ ,  $\tilde{C}_4 = \{3, 4, 6\}$  and  $\tilde{C}_5 = \{4, 5, 6\}$ . The maximal cliques of the chordal graph extension represent the subset of variables whose interaction is taken into account explicitly in the sparse relaxation. For instance, 1 and 3 are not included in a clique. This implies that there is no interaction between the variables  $x_1$  and  $x_3$  when any monomial in the objective polynomial function or any constraint polynomial function is evaluated.

We introduce a property characterizing a class of POPs for which the sparse SDP relaxation works effectively:

- (C) The csp graph  $G(N, E)$  has a sparse chordal extension such that the sizes of its maximal cliques are small.

We show that Example 18.2 satisfies this property. Suppose that  $n \geq 4$ . Then,  $F_k = \{k, k+1\}$  ( $k = 1, 2, \dots, n-1$ ), and the objective polynomial function  $f_0$  contains a monomial  $cx_1x_n$ . The csp graph  $G(N, E)$  consists of  $n$  edges  $(1, 2), (2, 3), \dots, (n-1, n), (n, 1)$ , which form a cycle. The graph  $G(N, E)$  is not chordal, but it has a sparse chordal extension  $G(N, \bar{E})$  shown in Fig. 18.4. And, the extended chordal graph  $G(N, \bar{E})$  consists of  $n-2$  cycles  $C_k = \{k, k+1, n\}$  ( $k = 1, 2, \dots, n-2$ ). Obviously,  $F_k \subset C_k$  ( $k = 1, 2, \dots, n-2$ ) and  $F_{n-1} \subset C_{n-2}$ , thus, we can take  $\tilde{C}_k = C_k$  ( $k = 1, 2, \dots, n-2$ ) and  $\tilde{C}_{n-1} = C_{n-2}$ .

In the next subsection, we discuss how the property (C) of a POP is transferred to its sparse SDP relaxation satisfying the properties (A) and (B).

### 18.3.4 Sparse SDP Relaxation of a POP

We use the following set  $\mathcal{A}_\psi^C$  instead of  $\mathcal{A}_\psi$  in the dense SDP relaxation. For every  $\psi \in \mathbb{Z}_+$  and every nonempty  $C \subset N = \{1, 2, \dots, n\}$ , define  $\mathcal{A}_\psi^C = \{\alpha \in \mathbb{Z}_+^n : \alpha_i = 0 \text{ } (i \notin C) \text{ and } \sum_{i \in C} \alpha_i \leq \psi\}$ , where  $\psi$  stands for either a relaxation order  $\omega$  or  $\omega_k - \omega$  ( $k = 1, 2, \dots, m$ ) and  $C \subset N$  a maximal clique of a chordal extension of the csp graph.

We choose a relaxation order  $\omega \geq \omega_{\max}$ . Using  $\omega_k$ ,  $\bar{C}_k$  ( $k = 1, 2, \dots, m$ ) and  $C_j$  ( $j = 1, 2, \dots, \ell$ ), we transform the POP (18.1) into an equivalent PSDP

$$\left. \begin{array}{l} \text{minimize } f_0(x) \\ \text{subject to } u(x, \mathcal{A}_{\omega-\omega_k}^{\bar{C}_k})u(x, \mathcal{A}_{\omega-\omega_k}^{\bar{C}_k})^T f_k(x) \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}^{\bar{C}_k}} \\ \quad (k = 1, 2, \dots, m), \\ u(x, \mathcal{A}_\omega^{C_j})u(x, \mathcal{A}_\omega^{C_j})^T \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_j}} \quad (j = 1, 2, \dots, \ell). \end{array} \right\} \quad (18.11)$$

Let

$$\mathcal{F}^s = \bigcup_{j=1}^{\ell} \left( \mathcal{A}_\omega^{C_j} + \mathcal{A}_\omega^{C_j} \right) = \bigcup_{j=1}^{\ell} \mathcal{A}_{2\omega}^{C_j}. \quad (18.12)$$

Then, we can rewrite the PSDP above as

$$\begin{aligned} & \text{minimize} \quad \sum_{\alpha \in \mathcal{F}^s} c_0^s(\alpha) x^\alpha \\ & \text{subject to} \quad \sum_{\alpha \in \mathcal{F}^s} L_k^s(\alpha, \omega) x^\alpha \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}^{\bar{C}_k}} \quad (k = 1, 2, \dots, m), \\ & \quad \sum_{\alpha \in \mathcal{F}^s} M_j^s(\alpha, \omega) x^\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_j}} \quad (j = 1, 2, \dots, \ell), \end{aligned}$$

for some  $c_0^s(\alpha) \in \mathbb{R}$  ( $\alpha \in \mathcal{F}^s$ ), real symmetric matrices  $L_k^s(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^s$ ,  $k = 1, 2, \dots, m$ ) and  $M_j^s(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^s$ ,  $j = 1, 2, \dots, \ell$ ). A sparse SDP relaxation problem of the POP (18.1) is obtained by replacing each monomial  $x^\alpha$  by a single real variable  $y_\alpha$ .

$$\left. \begin{array}{l} \text{minimize} \quad \sum_{\alpha \in \mathcal{F}^s} c_0^s(\alpha) y_\alpha \\ \text{subject to} \quad \sum_{\alpha \in \mathcal{F}^s} L_k^s(\alpha, \omega) y_\alpha \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}^{\bar{C}_k}} \quad (k = 1, \dots, m), \\ \quad \sum_{\alpha \in \mathcal{F}^s} M_j^s(\alpha, \omega) y_\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_j}} \quad (j = 1, \dots, \ell), \quad y_0 = 1. \end{array} \right\} \quad (18.13)$$

Let  $\zeta_\omega^s$  denote the optimal value of (18.13). Then,  $\zeta_\omega^s \leq \zeta_{\omega+1}^s \leq f_0^*$  and  $\zeta_\omega^s \leq \zeta_\omega^d$  for every  $\omega \geq \omega_{\max}$ , where  $f_0^*$  denotes the optimal value of (18.1) and  $\zeta_\omega^d$  the optimal value of the dense SDP relaxation problem (18.3). The second inequality  $\zeta_\omega^s \leq \zeta_\omega^d$  indicates that the sparse SDP relaxation is not always as strong as the dense SDP relaxation. The convergence of  $\zeta_\omega^s$  to  $f_0^*$  as  $\omega \rightarrow \infty$  was shown in [23] under a moderate condition similar to the one for the convergence of the dense SDP relaxation [22].

Let us apply the sparse SDP relaxation to Example 18.1 with  $\omega = \omega_{\max} = 1$ . Then the PSDP (18.2) is

$$\begin{aligned} & \text{minimize } x_2 - 2x_1x_2 + x_2x_3 \\ & \text{subject to } 1^2 \cdot (1 - x_1^2 - x_2^2) \geq 0, \quad 1^2 \cdot (1 - x_2^2 - x_3^2) \geq 0, \\ & \begin{pmatrix} x^{000} & x^{100} & x^{010} \\ x^{100} & x^{200} & x^{110} \\ x^{010} & x^{110} & x^{020} \end{pmatrix} = \begin{pmatrix} 1 & x_1 & x_2 \\ x_1 & x_1^2 & x_1x_2 \\ x_2 & x_1x_2 & x_2^2 \end{pmatrix} \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_1}}, \\ & \begin{pmatrix} x^{000} & x^{010} & x^{001} \\ x^{010} & x^{020} & x^{011} \\ x^{001} & x^{011} & x^{002} \end{pmatrix} = \begin{pmatrix} 1 & x_2 & x_3 \\ x_2 & x_2^2 & x_2x_3 \\ x_3 & x_2x_3 & x_3^2 \end{pmatrix} \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_2}}. \end{aligned}$$

Let

$$\mathcal{F}^s = \mathcal{A}_{2\omega}^{C_1} \bigcup \mathcal{A}_{2\omega}^{C_2} = \{(0,0,0), (1,0,0), (0,1,0), (2,0,0), (1,1,0), (0,2,0), (0,0,1), (0,1,1), (0,0,2)\}.$$

Replacing each  $x^\alpha$  by  $y_\alpha$  ( $\alpha \in \mathcal{F}^s$ ), we obtain an SDP relaxation problem

$$\begin{aligned} & \text{minimize } y_{010} - 2y_{110} + y_{011} \\ & \text{subject to } y_{000} - y_{200} - y_{020} \geq 0, \quad y_{000} - y_{020} - y_{002} \geq 0, \\ & \begin{pmatrix} y_{000} & y_{100} & y_{010} \\ y_{100} & y_{200} & y_{110} \\ y_{010} & y_{110} & y_{020} \end{pmatrix} \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_1}}, \quad \begin{pmatrix} y_{000} & y_{010} & y_{001} \\ y_{010} & y_{020} & y_{011} \\ y_{001} & y_{011} & y_{002} \end{pmatrix} \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_2}}, \\ & y_{000} = 1. \end{aligned}$$

We can rewrite this problems as

$$\begin{aligned} & \text{minimize } y_{010} - 2y_{110} + y_{011} \\ & \text{subject to } y_{000} - y_{200} - y_{020} \geq 0, \quad y_{000} - y_{020} - y_{002} \geq 0, \\ & \sum_{\alpha \in \mathcal{F}^s} M_1^s(\alpha, \omega) \alpha y_\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_1}}, \\ & \sum_{\alpha \in \mathcal{F}^s} M_2^s(\alpha, \omega) \alpha y_\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_2}}, \quad y_{000} = 1, \end{aligned}$$

for some  $M_j^s(\alpha, \omega) \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_j}}$  ( $\alpha \in \mathcal{F}^s$ ,  $j = 1, 2$ ); for example,

$$M_1^s((1,0,0), \omega) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad M_2^s((0,1,1), \omega) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

Assume that the POP (18.1) satisfies the property (C). We show how its sparse SDP relaxation problem inherits the property. Let  $G(N, E)$  be the csp graph and  $G(N, \bar{E})$  a chordal extension of  $G(N, E)$  with the maximal cliques  $C_1, C_2, \dots, C_\ell$ . Let  $R \in \mathbb{S}^{\mathcal{F}^s}$  be the csp matrix of the sparse SDP relaxation problem (18.13), and  $G(\mathcal{F}^s, \mathcal{E})$  the sparsity pattern graph of  $R$ , where  $\mathcal{F}^s$  is given by (18.12). By construction, we know that  $L_k^s(\alpha, \omega) = O$  if  $\alpha \notin \mathcal{A}_{2\omega}^{\tilde{C}_k}$  ( $k = 1, 2, \dots, m$ ),  $M_j^s(\alpha, \omega) = O$  if  $\alpha \notin \mathcal{A}_{2\omega}^{C_j}$  ( $j = 1, 2, \dots, \ell$ ) and that each  $\mathcal{A}_{2\omega}^{\tilde{C}_k}$  is contained in some  $\mathcal{A}_{2\omega}^{C_j}$ . Suppose that we construct a graph  $G(\mathcal{F}^s, \bar{\mathcal{E}})$  such that  $(\alpha, \beta) \in \bar{\mathcal{E}}$  if and only if  $(\alpha, \beta) \in \mathcal{A}_{2\omega}^{C_j}$  for some  $j = 1, 2, \dots, \ell$ . Then  $\mathcal{E} \subset \bar{\mathcal{E}}$ , i.e.,  $G(\mathcal{F}^s, \bar{\mathcal{E}})$  is an extension of  $G(\mathcal{F}^s, \mathcal{E})$ . Furthermore, we can prove that  $G(\mathcal{F}^s, \bar{\mathcal{E}})$  is a chordal graph with the maximal cliques  $\mathcal{A}_{2\omega}^{C_j}$  ( $j = 1, 2, \dots, \ell$ ). See Lemma 6.1 of [17]. As a result, the chordal extension  $G(\mathcal{F}^s, \bar{\mathcal{E}})$ , with the maximal clique  $\mathcal{A}_{2\omega}^{C_j}$  ( $j = 1, 2, \dots, \ell$ ), of the sparsity pattern graph of the sparse SDP relaxation problem (18.13) satisfies the same sparse structure as the chordal extension  $G(N, \bar{E})$ , with the maximal cliques  $C_j$ , ( $j = 1, 2, \dots, \ell$ ), of the csp graph  $G(N, E)$  of the POP (18.1). We note that the size of the former is larger than that of the latter. Consequently, if the maximal cliques  $C_j$  ( $j = 1, 2, \dots, \ell$ ) are small, the SDP relaxation problem (18.13) satisfies the properties (A) and (B). We illustrate this by applying the sparse SDP relaxation to Example 18.2.

For Example 18.2, the extended chordal graph  $G(N, \bar{E})$  of the csp graph consists of  $n - 2$  cycles  $C_k = \{k, k + 1, n\}$  ( $k = 1, 2, \dots, n - 2$ ). Hence, the size of the maximal cliques is 3. It follows that

$$\begin{aligned} \text{the size of } L_k^s(\alpha, \omega) &= \left| \mathcal{A}_{\omega-\omega_k}^{\tilde{C}_k} \right| = \binom{3 + \omega - \omega_k}{\omega - \omega_k} \\ &\quad (k = 1, 2, \dots, n - 1), \\ \text{the size of } M_j^s(\alpha, \omega) &= \left| \mathcal{A}_{\omega}^{C_j} \right| = \binom{3 + \omega}{\omega}, \\ &\quad (j = 1, 2, \dots, n - 2), \\ \text{the number of variables} &= |\mathcal{F}^s| \leq (n - 2) \binom{3 + 2\omega}{2\omega}, \end{aligned}$$

It should be noted that the sizes of  $L_k^s(\alpha, \omega)$  and  $M_j^s(\alpha, \omega)$  are independent of  $n$  and that the number of variables is linear in  $n$ . Table 18.2 shows numerical results on the sparse SDP relaxation applied to Example 18.2. Critical differences can be observed comparing Table 18.1 with Table 18.2. We mention that an accurate approximation to the optimal solution of the POP was computed for all cases of Table 18.2. More numerical results will be reported in Sect. 18.7.

**Table 18.2** Numerical results on the sparse SDP relaxation applied to Example 18.2. eTime denotes the elapsed time to solve the SDP problem by the Matlab version of SDPA 7.3.1 [29] on 3.06 GHz Intel Core 2 Duo with 8HB memory

$\gamma$	$n$	$\omega = \omega_{\max}$	Size $L_k^d$	Size $M^d$	$ \mathcal{F}^d $	eTime
2	10	1	1	4	38	0.03
2	20	1	1	4	78	0.04
2	40	1	1	4	158	0.08
4	10	2	4	10	175	0.06
4	20	2	4	10	375	0.13
4	40	2	4	10	775	0.28
6	10	3	10	20	476	0.22
6	20	3	10	20	1036	0.53
6	40	3	10	20	2156	1.23

## 18.4 Additional Techniques

We present important techniques used in SparsePOP [34] for its efficiency, accuracy and numerical stability, and for its application to practical problems.

### 18.4.1 Handling Equality, Lower Bound and Upper Bound Constraints

In practice, the sparse SDP relaxation described in Sect. 18.3.4 is applied to POPs with inequalities, equalities, lower bounds and/or upper bounds. More precisely, consider

$$\begin{aligned} & \text{minimize } f_0(x) \\ & \text{subject to } \left. \begin{array}{l} f_k(x) \geq 0 \quad (k = 1, 2, \dots, q), \\ f_k(x) = 0 \quad (k = q+1, \dots, m), \\ \lambda_i \leq x_i \leq \nu_i \quad (i \in N), \end{array} \right\} \end{aligned} \quad (18.14)$$

where  $-\infty \leq \lambda_i < \nu_i \leq \infty$  ( $i \in N$ ). We use the same notation and symbols as in Sect. 18.3.4. We first observe that the POP (18.14) is equivalent to the PSDP

$$\begin{aligned} & \text{minimize } f_0(x) \\ & \text{subject to } \left. \begin{array}{l} u(x, \mathcal{A}_{\omega-\omega_k}^{\tilde{C}_k}) u(x, \mathcal{A}_{\omega-\omega_k}^{\tilde{C}_k})^T f_k(x) \in \mathbb{S}_+^{\tilde{C}_k} \\ \quad (k = 1, 2, \dots, q), \\ u(x, \mathcal{A}_{\omega-\omega_k}^{\tilde{C}_k}) u(x, \mathcal{A}_{\omega-\omega_k}^{\tilde{C}_k})^T f_k(x) = O \\ \quad (k = q+1, q+2, \dots, m), \end{array} \right. \end{aligned}$$

$$u(x, \mathcal{A}_\omega^{C_j}) u(x, \mathcal{A}_\omega^{C_j})^T \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_j}} \quad (j = 1, 2, \dots, \ell),$$

$$\lambda_i \leq x_i \leq v_i \quad (i \in N),$$

where the relaxation  $\omega$  is chosen such that  $\omega \geq \omega_{\max} = \max\{\omega_k : k = 0, 1, \dots, m\}$ . We then rewrite this problem as

$$\begin{aligned} & \text{minimize} \quad \sum_{\alpha \in \mathcal{F}^s} c_0^s(\alpha) x^\alpha \\ & \text{subject to} \quad \sum_{\alpha \in \mathcal{F}^s} L_k^s(\alpha, \omega) x^\alpha \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}^{\bar{C}_k}} \quad (k = 1, 2, \dots, q), \\ & \quad \sum_{\alpha \in \mathcal{F}^s} L_k^s(\alpha, \omega) x^\alpha = O \quad (k = q+1, q+2, \dots, m), \\ & \quad \sum_{\alpha \in \mathcal{F}^s} M_j^s(\alpha, \omega) x^\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_j}} \quad (j = 1, 2, \dots, \ell), \\ & \quad \lambda_i \leq x_i \leq v_i \quad (i \in N), \end{aligned}$$

for some  $c_0^s(\alpha) \in \mathbb{R}$  ( $\alpha \in \mathcal{F}^s$ ), real symmetric matrices  $L_k^s(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^s$ ,  $k = 1, 2, \dots, m$ ) and  $M_j^s(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^s$ ,  $j = 1, 2, \dots, \ell$ ). Thus, replacing each monomial  $x^\alpha$  by a single real variable  $y_\alpha$ , we obtain the SDP relaxation problem of the POP (18.14)

$$\left. \begin{aligned} & \text{minimize} \quad \sum_{\alpha \in \mathcal{F}^s} c_0^s(\alpha) y_\alpha \\ & \text{subject to} \quad \sum_{\alpha \in \mathcal{F}^s} L_k^s(\alpha, \omega) y_\alpha \in \mathbb{S}_+^{\mathcal{A}_{\omega-\omega_k}^{\bar{C}_k}} \quad (k = 1, 2, \dots, q), \\ & \quad \sum_{\alpha \in \mathcal{F}^s} L_k^s(\alpha, \omega) y_\alpha = O \quad (k = q+1, q+2, \dots, m), \\ & \quad \sum_{\alpha \in \mathcal{F}^s} M_j^s(\alpha, \omega) y_\alpha \in \mathbb{S}_+^{\mathcal{A}_\omega^{C_j}} \quad (j = 1, 2, \dots, \ell), \quad y_0 = 1, \\ & \quad \lambda_i \leq y_{e^i} \leq v_i \quad (i \in N), \end{aligned} \right\} \quad (18.15)$$

where  $e^i$  denotes the  $i$ th unit vector in  $\mathbb{R}^n$  ( $i \in N$ ). For each  $k = q+1, q+2, \dots, m$ , all coefficient matrices  $L_k^s(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^s$ ) of the equality constraint  $\sum_{\alpha \in \mathcal{F}^s} L_k^s(\alpha, \omega) y_\alpha = O \in \mathbb{S}^{\mathcal{A}_{\omega-\omega_k}^{\bar{C}_k}}$  are symmetric. Hence, we can rewrite this equality constraint with respect to each component in the lower triangular part of  $L_k^s(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^s$ ).

In the above derivation of the sparse SDP relaxation (18.15), we have treated the lower and upper bound constraints of the POP (18.14) differently from the inequality constraints  $f_k(x) \geq 0$  ( $k = 1, 2, \dots, m$ ). Adding bound constraints is mainly

for numerical stability in solving the sparse SDP relaxation. We could treat the bound constraints in the same way as the inequality constraints to guarantee the theoretical convergence, but then the resulting SDP relaxation would be larger-sized and more expensive to solve.

### 18.4.2 Computing Optimal Solutions

After solving the sparse SDP relaxation by an SDP solver, an approximation to an optimal solution of the POP (18.14) needs to be extracted. We describe a simple method used in SparsePOP. By default, SparsePOP assumes that the POP (18.14) to be solved has a unique optimal solution. Let  $\zeta_\omega$  denote the optimal objective value and  $(y_a^\omega : \alpha \in \mathcal{F}^S)$  the optimal solution of the SDP (18.15). Note that the values of  $y_a^\omega$  ( $\alpha \in \{e^i : i \in N\}$ ) correspond to the variable  $x_i$  ( $i \in N$ ) in the POP (18.14). Thus, these values can be used as an approximation to an optimal solution  $x^\omega$  of the POP (18.14). Let  $x_i^\omega = y_{e^i}$  ( $i \in N$ ) and  $x^\omega = (x_1^\omega, x_2^\omega, \dots, x_n^\omega)$ . We know that  $f_0(x) \geq f_0^* \geq \zeta_\omega$  for every feasible solution  $x$  of the POP (18.14), where  $f_0^*$  denotes the unknown optimal value of (18.14). Therefore, we may regard  $x^\omega$  an approximation to the optimal solution of (18.14) (a) if  $x^\omega$  (approximately) satisfies the constraints of (18.14) and (b) if  $f_0(x^\omega) - \zeta_\omega$  is sufficiently small. SparsePOP provides output information to decide whether (a) and (b) are satisfied.

If the POP (18.14) has multiple global optimal solutions, the method described previously will not work. In this case, SparsePOP replaces the objective polynomial  $f_0(x)$  by  $f_0(x) + \epsilon d^T x$  with an  $n$ -dimensional column vector  $d$  whose components are chosen randomly from  $[0, 1]$  and  $\epsilon > 0$  a small positive parameter controlling the magnitude of the perturbation term  $\epsilon d^T x$ . The POP with this perturbed objective function is expected to have a unique optimal solution. Then, we can apply the method described previously, and we may regard  $x^\omega$  an approximation to an optimal solution of the POP (18.14) if (a) and (b) are satisfied. See the paper [34] for more details.

A linear algebra method proposed by Henrion and Lasserre [10] computes multiple optimal solutions of the POP (18.1) from an optimal solution of the dense SDP relaxation problem (18.3). This method was extended to the sparse SDP relaxation problem (18.13) [23]. It was not implemented in SparsePOP since the cost of solving large-scale POPs is expected to be much more expensive than the simple method discussed in this subsection.

### 18.4.3 Choosing a Higher Relaxation Order

If no parameter is specified, SparsePOP applies the sparse SDP relaxation with the relaxation order  $\omega = \omega_{\max}$  to a POP to be solved. If the obtained solution is not within the range of desired accuracy, one can run SparsePOP again with

different values of the parameters to attain an optimal solution of higher accuracy. In particular, the relaxation order  $\omega$  determines both the quality of an approximate solution and the size of the SDP relaxation problem (18.15) of the POP (18.14). An approximation to an optimal solution of the POP (18.14) with higher accuracy or not lower accuracy is expected by solving the SDP relaxation problem (18.15) as a larger value is chosen for  $\omega$ . This, however, increases the cost of solving the SDP relaxation (18.15). Thus,  $\omega = \omega_{\max}$  is used initially, and it is successively increased by 1 if the approximate solution  $x^\omega$  with higher accuracy needs to be found.

#### 18.4.4 Scaling

In many POPs from applications, it is necessary to perform a scaling to obtain meaningful numerical solutions. The scaling technique described here is intended to improve the numerical stability. Suppose that both of the lower and upper bounds on  $x_i$  are finite, i.e.,  $-\infty < \lambda_i < v_i < \infty$  ( $i \in N$ ) in the POP (18.14). We perform a linear transformation to the variables  $x_i$  such that  $z_i = (x_i - \lambda_i)/(v_i - \lambda_i)$ . The objective and constrained polynomials  $g_k \in \mathbb{R}[z]$  ( $k = 0, 1, \dots, m$ ) become

$$\begin{aligned} g_k(z_1, z_2, \dots, z_n) \\ = f_k((v_1 - \lambda_1)z_1 + \lambda_1, (v_2 - \lambda_2)z_2 + \lambda_2, \dots, (v_n - \lambda_n)z_n + \lambda_n). \end{aligned}$$

Then, normalize each  $g_k \in \mathbb{R}[z]$  such that  $h_k(z) = g_k(z)/\chi_k$ , where  $\chi_k$  denotes the maximum of the magnitude of the coefficients of the polynomial  $g_k \in \mathbb{R}[z]$  ( $k = 0, 1, 2, \dots, m$ ). Consequently, we obtain a scaled POP

$$\begin{aligned} & \text{minimize } h_0(z) \\ & \text{subject to } h_k(z) \geq 0 \ (k = 1, 2, \dots, q), \\ & \quad h_k(z) = 0 \ (k = q+1, \dots, m), \ 0 \leq z_i \leq 1 \ (i \in N), \end{aligned}$$

which is equivalent to the POP (18.14).

#### 18.4.5 Reducing the Sizes of SDP Relaxation Problems

A method to exploit the sparsity of SOS polynomials was proposed in [4] to reduce the size of the SDP relaxation. This method is implemented in SparsePOP to reduce the sizes of the coefficient matrices  $M_j^s(\alpha, \omega)$  ( $\alpha \in \mathcal{F}^s$ ,  $j = 1, 2, \dots, \ell$ ) in the sparse SDP relaxation (18.15) of the POP (18.14). See [19] for details.

## 18.5 Numerical Results on SparsePOP with SDPA

SparsePOP [31] is a Matlab package for solving unconstrained and constrained POPs of the form (18.14) by the sparse (or dense) SDP relaxation method. When SparsePOP is called for a POP specified in either the SparsePOP format [31] or the GAMS scalar format [7], a sparse (or dense) SDP relaxation problem is first constructed with given parameters, and then solved by the SDP solver SeDuMi [32] or SDPA [29, 36]. At the end of computation, various information including approximations to the global optimal value and solution, a lower bound for the global optimal value of the POP and the elapsed time for solving the SDP problem are provided. As an option, SparsePOP refines the computed approximation to the global optimal solution by applying the Matlab functions fminunc, fmincon or lsqnonlin in Matlab Optimization Toolbox.

We compared the dense and sparse SDP relaxations, which were implemented in SparsePOP 2.20, with selected test problems from [5, 8, 26] to confirm the effectiveness of exploiting the sparsity of POPs for improving the computational efficiency. We used a Matlab version of SDPA 7.3.1 as an SDP solver, and performed all numerical experiments on a 2.8 GHz Intel Quad-Core i7 with 16 GB memory.

The numerical results are shown in Table 18.3. The chained singular function, the Broyden tridiagonal function, the Rosenbrock function, and the chained wood function in Table 18.3 are unconstrained problems of the form: minimize  $f_0(x)$  over  $x \in \mathbb{R}^n$ . The other test problems are constrained POPs of the form (18.14) from [8]. The Broyden tridiagonal function has two global minimizers, the one with  $x_1 > 0$  and the other with  $x_1 < 0$ . We added the constraint  $x_1 \geq 0$  to exclude the second solution so that the function has a unique global optimal solution subject to the constraint. For the numerical stability, we added lower and upper bounds for the variables of some constrained POPs. See Sect. 18.4.4. The following is the notation for Table 18.3.

eTime = the elapsed time in SDPA (the elapsed time

in the Matlab function fmincon) in seconds,

$$rObjErr = \frac{|\text{opt. val. of SDP} - f_0(\hat{x})|}{\max\{1, |f_0(\hat{x})|\}},$$

$$\text{absErr} = \min\{f_j(\hat{x}) \ (j = 1, 2, \dots, q), -|f_k(\hat{x})| \ (k = q + 1, \dots, m)\},$$

where  $\hat{x}$  denotes an approximation to the optimal solution.

In Table 18.3, we observe that the size of unconstrained POPs that could be handled by the sparse SDP relaxation is much larger than that of the dense SDP relaxation. More precisely, the sparse SDP relaxation provided an accurate solution for the unconstrained problems with  $n = 10,000$  while the dense SDP relaxation solved the problems of size up to  $n = 24$ . For the constrained problems, the elapsed time for solving the sparse SDP relaxation problem is much shorter than that for solving the dense SDP relaxation problem. We can see that exploiting the sparsity

**Table 18.3** Numerical results on the dense and sparse relaxations applied to unconstrained and constrained POPs

Problem	n	Dense SDP (+finincon)				Sparse SDP (+finincon)			
		$\omega_{\max}$	$\omega$	eTime	rObjErr	eTime	rObjErr	absErr	absErr
Chained Singular	12	2	2	8.7 6281.7	4.6e-05 2.2e-05	0.1	6.9e-04		
Broyden Tridiag.	24	2	2			0.1	3.3e-04		
	1000	2	2			6.2	8.8e-04		
	10000	2	2			79.1	5.8e-04		
	12	2	2	8.4 6364.9	1.6e-06 7.8e-07	0.1	5.7e-07		
	24	2	2			0.1	1.2e-06		
	1000	2	2			5.6	4.3e-06		
	10000	2	2			61.8	9.2e-04		
Chained Wood	12	2	2	0.2 30.9	7.9e-06 7.3e-07	0.0	5.1e-05		
	24	2	2			0.0	1.0e-05		
	1000	2	2			1.5	4.4e-04		
	10000	2	2			20.7	4.4e-03		
Rosenbrock	12	2	2	4.6 3974.7	2.2e-06 8.6e-07	0.0	8.2e-05		
	24	2	2			0.1	9.4e-05		
	1000	2	2			3.3	6.0e-05		
	10000	2	2			37.2	7.2e-05		
ex2_1.8	24	1	1	0.2(0.4) 34.2(0.0)	6.5e00 1.2e-06	0.1(0.0) 4.6(0.0)	6.5e00 2.7e-06		
				2 0.3(0.2)	3.3e-02 4.8e-07	-9.4e-13 0.1(0.1)	-1.7e-16 3.5e-02	-1.8e-16 -1.8e-14	
ex3_1_1	8	1	2			-7.0e-14	4.3e-07	-6.2e-14	
ex5_2_2	9	1	3	72.8(0.0)	5.7e02 2.0e-04	-1.4e-12 0.0(0.1)	5.7e02 -2.5e-09	-1.4e-12 0.1(0.1)	
_case1				0.0(0.1) 0.3(0.0)	1.8e-04 1.8e-04	0.8(0.0) -2.5e-12	1.3e-01 0.8(0.0)	1.3e-01 1.8e-03	
ex5_3_2	22	1	1	0.0(4.3) 43.2(0.1)	4.6e-01 5.0e-06	-6.7e-17 -1.5e-09	14.5(0.9) 0.9(5.7)	4.6e-01 1.5e-01	
				3		48.1(1.1)		-9.1e-17 -3.3e-14	
ex5_4_2	8	1	2	0.4(0.9) 65.7(0.0)	5.1e-01 1.1e-05	-2.0e-16 -1.9e-16	0.1(0.1) 0.5(0.0)	5.2e-01 5.3e-08	
alkyl	14	2	2	2.8(0.3) 89.5(3.0)	1.0e-01 1.4e-05	-1.3e-08 -3.6e-13	0.1(0.1) 0.9(0.0)	1.5e-01 8.2e-06	
				3				-1.1e-07 -1.1e-07	

was crucial to reduce the elapsed time. The errors in the sparse SDP relaxation are compatible to those in the dense SDP relaxation.

Important factors that affect the computational efficiency for solving SDPs are:

- no = the number of positive semidefinite matrix variables  
of the SDP relaxation problem in the SDPA sparse format,
- max = the maximum size of positive semidefinite matrix variables  
of the SDP relaxation problem in the SDPA sparse format,
- sizeL = the size of the Schur complement matrix,
- nnzL = the number of nonzeros in a sparse Cholesky factor  
of the Schur complement matrix.

Table 18.4 shows these numbers for the dense and sparse SDP relaxations. We notice that the number of positive semidefinite variable matrices of the sparse SDP relaxation is larger than that of the dense SDP relaxation. The maximum size of positive semidefinite matrix variables coincides with the size of  $\mathcal{A}_\omega$  in the dense SDP relaxation problem (18.3), while it coincides with the maximum of sizes of  $\mathcal{A}_\omega^{C_j}$  ( $j = 1, 2, \dots, \ell$ ) in the sparse SDP relaxation problem (18.13). We observe that the difference becomes larger as  $n$  or  $\omega$  increases. Notice that the size and the number of nonzeros of the Cholesky factor  $L$  of the Schur complement matrix  $B$  are both much smaller in the sparse SDP relaxation. This confirms that these factors considerably contributed to improving the efficiency of the sparse SDP relaxation.

## 18.6 An Application to the Sensor Network Localization Problem

We consider a sensor network localization (SNL) problem of  $n$  sensors in  $\mathbb{R}^s$ : Compute locations of sensors when distances between pairs of sensors located closely are available and locations of some of the sensors are provided. We present the full SDP (FSDP) relaxation [2] of the SNL problem and its sparse variant SFSDP relaxation [15] using the frameworks of the dense SDP relaxation in Sect. 18.2 and the sparse SDP relaxation in Sect. 18.3, respectively.

### 18.6.1 Quadratic Optimization Formulation of the SNL Problem

We assume that the location  $a_r \in \mathbb{R}^s$  of sensor  $r$  is known for  $r = m + 1, \dots, n$ . These sensors are called *anchors* in the subsequent discussion. We denote the number of anchors by  $m_a (= n - m)$ . Let  $\rho > 0$  be a radio range, which determines the set  $\mathcal{N}_x^\rho$  for

**Table 18.4** Comparison of the dense and sparse SDP relaxations

Problem	n	$\omega$	Dense SDP				Sparse SDP			
			Mat. var.		$B = LL^T$		Mat. var.		$B = LL^T$	
			no	max	sizeL	nnzL	no	max	sizeL	nnzL
Broyden	12	2	2	91	1819	1655290	11	10	214	5005
Tridiag.	24	2	2	325	20474	209602575	23	10	454	10885
	1000	2					999	10	19974	489125
	10000	2					9999	10	199974	4899125
Chained	12	2	1	34	398	79401	11	4	53	261
Wood	24	2	1	103	2989	4468555	23	4	107	531
	1000	2					999	4	4499	22491
	10000	2					9999	4	44999	224991
ex5_4_2	8	2	23	14	320	36315	25	7	97	1929
	3	23	83	2459	3023718	25	22	310	20496	
alkyl	14	2	29	32	1181	308751	38	10	203	3886
	3	29	264	22071	125984806	38	28	834	64729	

pairs of sensors  $p$  and  $q$  such that their (Euclidean) distance  $d_{pq}$  does not exceed  $\rho$ , and the set  $\mathcal{N}_a^\rho$  for pairs of a sensor  $p$  and an anchor  $r$  such that their distance  $d_{pr}$  does not exceed  $\rho$ ;

$$\mathcal{N}_x^\rho = \{(p, q) : 1 \leq p < q \leq m, \|x_p - x_q\| \leq \rho\},$$

$$\mathcal{N}_a^\rho = \{(p, r) : 1 \leq p \leq m, m+1 \leq r \leq n, \|x_p - a_r\| \leq \rho\},$$

where  $x_p \in \mathbb{R}^s$  denotes the unknown location of sensor  $p$  ( $p = 1, 2, \dots, m$ ). Let  $\mathcal{N}_x$  be a subset of  $\mathcal{N}_x^\rho$  and  $\mathcal{N}_a$  a subset of  $\mathcal{N}_a^\rho$ . By introducing zero objective function and the distance equations as constraints, we have the following form of SNL problem with exact distance:

$$\begin{aligned} & \text{minimize } 0 \\ & \text{subject to } \|x_p - x_q\|^2 = d_{pq}^2 \quad (p, q) \in \mathcal{N}_x, \\ & \quad \|x_p - a_r\|^2 = d_{pr}^2 \quad (p, r) \in \mathcal{N}_a, \end{aligned}$$

or equivalently,

$$\begin{aligned} & \text{minimize } 0 \\ & \text{subject to } \left. \begin{aligned} & x_p^T x_p - 2x_p^T x_q + x_q^T x_q = d_{pq}^2 \quad (p, q) \in \mathcal{N}_x, \\ & x_p^T x_p - 2a_r^T x_p + a_r^T a_r = d_{pr}^2 \quad (p, r) \in \mathcal{N}_a. \end{aligned} \right\} \end{aligned} \quad (18.16)$$

For problems with noise, we consider

$$\begin{aligned} & \text{minimize } \sum_{(p, q) \in \mathcal{N}_x} (\xi_{pq}^+ + \xi_{pq}^-) + \sum_{(p, r) \in \mathcal{N}_a} (\xi_{pr}^+ + \xi_{pr}^-) \\ & \text{subject to } \left. \begin{aligned} & x_p^T x_p - 2x_p^T x_q + x_q^T x_q + \xi_{pq}^+ - \xi_{pq}^- = \hat{d}_{pq}^2 \quad (p, q) \in \mathcal{N}_x, \\ & x_p^T x_p - 2a_r^T x_p + a_r^T a_r + \xi_{pr}^+ - \xi_{pr}^- = \hat{d}_{pr}^2 \quad (p, r) \in \mathcal{N}_a, \\ & \xi_{pq}^+ \geq 0, \xi_{pq}^- \geq 0 \quad (p, q) \in \mathcal{N}_x, \\ & \xi_{pr}^+ \geq 0, \xi_{pr}^- \geq 0 \quad (p, r) \in \mathcal{N}_a, \end{aligned} \right\} \end{aligned} \quad (18.17)$$

where  $\xi_{pq}^+ + \xi_{pq}^-$  (or  $\xi_{pr}^+ + \xi_{pr}^-$ ) indicates 1-norm error in the square of estimated distance  $\hat{d}_{pq}$  between sensors  $p$  and  $q$  (or estimated distance  $\hat{d}_{pr}$  between sensor  $p$  and anchor  $r$ ).

### 18.6.2 Dense SDP Relaxation

We apply the dense SDP relaxation described in Sect. 18.2 to (18.16). The resulting SDP relaxation (18.18) coincides with FSDP relaxation proposed in the paper [2]. Obviously,  $\omega_{\max} = 1$ . Let  $\omega = \omega_{\max}$ . Then, the QOP (18.16) is transformed to an

equivalent quadratic SDP (QSDP) as the POP (18.1) transformed to the PSDP (18.2). Notice in (18.16) that each vector variable  $x_p$  appears in the inner products but its elements  $x_{pi}$  ( $i = 1, 2, \dots, s$ ) do not appear explicitly. Using this observation, we can modify the construction of an QSDP equivalent to (18.16). We consider the QSDP

$$\begin{aligned} & \text{minimize} \quad 0 \\ & \text{subject to} \quad x_p^T x_p - 2x_p^T x_q + x_q^T x_q - d_{pq}^2 = 0 \quad (p, q) \in \mathcal{N}_x, \\ & \quad x_p^T x_p - 2a_r^T x_p + a_r^T a_r - d_{pr}^2 = 0 \quad (p, r) \in \mathcal{N}_a, \\ & \quad (I, x_1, x_2, \dots, x_m)^T (I, x_1, x_2, \dots, x_m) \succeq O, \end{aligned}$$

which is equivalent to (18.16). Here  $I$  denotes the  $s \times s$  identity matrix. Define an  $s \times m$  matrix variable  $X = (x_1, x_2, \dots, x_m)$ . Then the last positive semidefinite constraint can be rewritten as  $\begin{pmatrix} I & X \\ X^T & X^T X \end{pmatrix} \succeq O$ . Replacing the quadratic term  $X^T X$  by a matrix variable  $Y \in \mathbb{S}^m$  leads to the FSDP relaxation [2] of (18.16).

$$\left. \begin{aligned} & \text{minimize} \quad 0 \\ & \text{subject to} \quad Y_{pp} - 2Y_{pq} + Y_{qq} = d_{pq}^2 \quad (p, q) \in \mathcal{N}_x, \\ & \quad Y_{pp} - 2a_r^T X_{\cdot p} + a_r^T a_r = d_{pr}^2 \quad (p, r) \in \mathcal{N}_a, \\ & \quad \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} \succeq O, \end{aligned} \right\} \quad (18.18)$$

where  $X_{\cdot p}$  denotes the  $p$ th column of  $X$ , i.e.,  $X_{\cdot p} = x_p$  ( $p = 1, 2, \dots, m$ ).

Similarly, we obtain the dense SDP relaxation of the SNL problem (18.17) with noise.

$$\begin{aligned} & \text{minimize} \quad \sum_{(p,q) \in \mathcal{N}_x} (\xi_{pq}^+ + \xi_{pq}^-) + \sum_{(p,r) \in \mathcal{N}_a} (\xi_{pr}^+ + \xi_{pr}^-) \\ & \text{subject to} \quad Y_{pp} - 2Y_{pq} + Y_{qq} + \xi_{pq}^+ - \xi_{pq}^- = \hat{d}_{pq}^2 \quad (p, q) \in \mathcal{N}_x, \\ & \quad Y_{pp} - 2a_r^T X_{\cdot p} + a_r^T a_r + \xi_{pr}^+ - \xi_{pr}^- = \hat{d}_{pr}^2 \quad (p, r) \in \mathcal{N}_a, \\ & \quad \xi_{pq}^+ \geq 0, \quad \xi_{pq}^- \geq 0 \quad (p, q) \in \mathcal{N}_x, \\ & \quad \xi_{pr}^+ \geq 0, \quad \xi_{pr}^- \geq 0 \quad (p, r) \in \mathcal{N}_a, \\ & \quad \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} \succeq O. \end{aligned}$$

### 18.6.3 Sparse SDP Relaxation

We apply the sparse SDP relaxation described in Sect. 18.3 to (18.16). A chordal-graph structured sparsity is extracted from (18.16) using the fact that the expression of (18.16) includes each vector variable  $x_p$  in the inner products, not individual

element  $x_{pi}$  ( $i = 1, 2, \dots, s$ ). Specifically, each vector variable  $x_p$  is regarded as a single variable when the csp graph  $G(V, E)$  of (18.16) is constructed. Let  $V = \{1, 2, \dots, m\}$  and

$$E = \left\{ (p, q) \in V \times V : \begin{array}{l} p < q, x_p \text{ and } x_q \text{ are involved} \\ \text{in an equality constraint of (18.16).} \end{array} \right\}$$

By construction, we know that  $E = \mathcal{N}_x$ . Let  $G(V, \bar{E})$  be a chordal extension of  $G(V, E)$  and  $C_j$  ( $j = 1, 2, \dots, \ell$ ) its maximal cliques. Then, consider the QSDP

$$\begin{aligned} & \text{minimize} \quad 0 \\ & \text{subject to} \quad \begin{cases} x_p^T x_p - 2x_p^T x_q + x_q^T x_q - d_{pq}^2 = 0 & (p, q) \in \mathcal{N}_x, \\ x_p^T x_p - 2a_r^T x_p + a_r^T a_r - d_{pr}^2 = 0 & (p, r) \in \mathcal{N}_a, \\ \begin{pmatrix} I & X_{C_j} \\ X_{C_j}^T & X_{C_j}^T X_{C_j} \end{pmatrix} \succeq O & (j = 1, 2, \dots, \ell), \end{cases} \end{aligned} \quad (18.19)$$

which is equivalent to (18.16). Here  $X_{C_j}$  denotes the submatrix of  $X$  consisting of column vectors  $x_p$  ( $p \in C_j$ ). We replace every  $x_p^T x_q$  in (18.19) by a real variable  $Y_{pq}$  and define a matrix variable  $Y_{C_j C_j}$  of  $Y_{pq}$  ( $(p, q) \in C_j \times C_j$ ) ( $j = 1, 2, \dots, \ell$ ). Then, the resulting SDP is

$$\begin{aligned} & \text{minimize} \quad 0 \\ & \text{subject to} \quad \begin{cases} Y_{pp} - 2Y_{pq} + Y_{qq} - d_{pq}^2 = 0 & (p, q) \in \mathcal{N}_x, \\ Y_{pp} - 2a_r^T X_{\cdot p} + a_r^T a_r - d_{pr}^2 = 0 & (p, r) \in \mathcal{N}_a, \\ \begin{pmatrix} I & X_{C_j} \\ X_{C_j}^T & Y_{C_j C_j} \end{pmatrix} \succeq O & (j = 1, 2, \dots, \ell). \end{cases} \end{aligned} \quad (18.20)$$

The sparse SDP relaxation problem (18.20) is exactly the same as SFSDP relaxation problem proposed in Sect. 18.3.3 of [15] for a sparse variant of FSDP [2], although the derivation of SFSDP from FSDP there is different. It was also shown in [15] that SFSDP (i.e., (18.20)) is equivalent to FSDP (i.e., (18.18)) in the sense that their feasible solution sets coincide with each other.

In the sparse SDP relaxation problem (18.20), we usually have  $C_j \cap C_k \neq \emptyset$  for some distinct  $j$  and  $k$ . Thus, two positive semidefinite constraints

$$\begin{pmatrix} I & X_{C_j} \\ X_{C_j}^T & Y_{C_j C_j} \end{pmatrix} \succeq O \quad \text{and} \quad \begin{pmatrix} I & X_{C_k} \\ X_{C_k}^T & Y_{C_k C_k} \end{pmatrix} \succeq O$$

share some variables  $X_{ip}$  ( $i = 1, 2, \dots, s$ ,  $p \in C_j \cap C_k$ ) and  $Y_{pq}$  ( $p \in C_j \cap C_k$ ,  $q \in C_j \cap C_k$ ). Thus, the sparse SDP problem (18.20) is not a standard SDP. It should be converted to an equality standard form or an LMI standard form of SDP to apply the primal-dual interior-point method [3, 29, 32, 33, 36]. A simple method is to represent

each matrix  $\begin{pmatrix} O & X_{C_j} \\ X_{C_j}^T & Y_{C_j C_j} \end{pmatrix}$  as a linear combination of some constant matrices with

the variables  $X_{ip}$  ( $i = 1, 2, \dots, s$ ,  $p \in C_j$ ) and  $Y_{pq}$  ( $(p, q) \in C_j \times C_j$ ,  $p \leq q$ ) in the matrix. See [13] for more details on such conversions.

Although how  $\mathcal{N}_x$  is selected from  $\mathcal{N}_x^\rho$  and  $\mathcal{N}_a$  from  $\mathcal{N}_a^\rho$  is not mentioned, it is a very important issue since their choice determines both the chordal-graph structured sparsity and the quality of the sparse SDP relaxation. As more edges from  $\mathcal{N}_x^\rho$  are chosen for  $\mathcal{N}_x$ , the csp graph  $G(V, E)$  becomes denser. (Recall that  $E = \mathcal{N}_x$ .) Conversely, if not enough edges from  $\mathcal{N}_x^\rho$  are chosen for  $\mathcal{N}_x$ , then the quality of the resulting sparse SDP relaxation would be deteriorated. For details, we refer to [15, 16].

#### 18.6.4 Numerical Results on SFSDP

We report numerical results on the software package SFSDP [16, 30], which is a Matlab implementation of the sparse SDP relaxation in the previous subsection. The package also includes the dense SDP relaxation in Sect. 2.2. We used a Matlab version of SDPA 7.3.1 as an SDP solver, and performed all numerical experiments on a 2.8 GHz Intel Quad-Core i7 with 16 GB memory. SNL problems with 1000 to 5000 sensors in  $\mathbb{R}^3$  were used for numerical experiments. Sensors and anchors were distributed randomly in the unit cube  $[0, 1]^3$ . The number of anchors was 10% or 5% of the number of sensors. The noisy factor  $\sigma$  was changed from 0.0 to 0.2, the radio range  $\rho$  was fixed to 0.25, and the distances were perturbed to create a noisy problem such that  $\hat{d}_{pq} = \max\{(1 + \sigma\epsilon_{pq}), 0.1\}\|a_p - a_q\|$  ( $(p, q) \in \mathcal{N}_x^\rho$ ) and  $\hat{d}_{pr} = \max\{(1 + \sigma\epsilon_{pr}), 0.1\}\|a_p - a_r\|$  ( $(p, r) \in \mathcal{N}_a^\rho$ ), where  $\epsilon_{pq}$  and  $\epsilon_{pr}$  were chosen from the standard normal distribution  $N(0, 1)$ , and  $a_p$  denotes the true location of sensor  $p$ . To measure the accuracy of locations of  $m$  sensors computed by SDPA, and the accuracy of refined solutions by the gradient method [1, 25], we used the root mean square distance (RMSD)  $\left(\frac{1}{m} \sum_{p=1}^m \|x_p - a_p\|^2\right)^{1/2}$ , where  $x_p$  denotes the computed locations of sensor  $p$ .

Table 18.5 shows the performance of the dense and sparse SDP relaxations for solving SNL problems with 1000 sensors and 100 randomly distributed anchors. The noisy factor  $\sigma$  was changed from 0.0 to 0.2. SDPA eTime denotes the elapsed time by SDPA. We see that the sparse SDP relaxation solves the problems much faster than the dense SDP relaxation while achieving compatible accuracy as indicated in RMSD. The left figure of Fig. 18.5 shows the locations of anchors  $\diamond$ , the true locations of sensors  $\circ$  and the computed locations of sensors  $\star$  for the problem with 1000 sensors, 100 anchors,  $\sigma = 0.2$  and  $\rho = 0.25$ . We also observe that the maximum size of matrix variables and the number of nonzeros of the Cholesky factor  $L$  of the Schur complement matrix  $B$  are much smaller in the sparse SDP relaxation than those in the dense SDP relaxation. This results in much faster elapsed time by the sparse SDP relaxation. See also the right figure of Fig. 18.5.

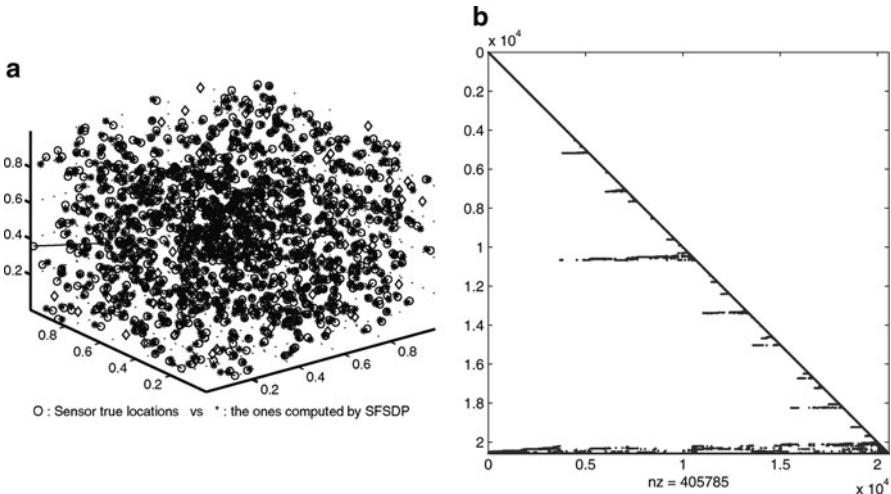
We further tested the sparse SDP relaxation for SNL problems of 3000 and 5000 sensors, and showed the results in Table 18.6. The elapsed time by SDPA for solving the resulting SDPs remains short, obtaining accurate values of RMSD. We confirm that exploiting sparsity greatly reduces elapsed time.

**Table 18.5** Comparison of the dense and sparse SDP relaxations to solve 3-dimensional problems with 1000 sensors, 100 anchors, and  $\rho = 0.25$ 

Test problems	$m, m_a, \rho$	$\sigma$	SDP		$B = LL^T$		RMSD		SDPA	
			relaxation	Mat. var.	sizeL	nnzL	SDP	w.Grad.	eTime	
$m = 1000,$	0.0	Dense	1	1003	5554	30846916	5.2e-4	3.1e-5	189.7	
$m_a = 100$		Sparse	962	27	7234	337084	2.3e-5	6.3e-6	5.7	
distributed	0.1	Dense	1	1003	20034	401361156	5.5e-2	9.3e-3	744.1	
randomly		Sparse	962	27	20586	405785	5.5e-2	9.3e-3	7.1	
$\rho = 0.25$	0.2	Dense	1	1003	20034	401361156	8.0e-2	2.2e-2	860.1	
		Sparse	962	27	20586	405785	8.0e-2	2.2e-2	7.3	

**Table 18.6** The sparse SDP relaxation to solve 3-dimensional SNL problems with 3000 sensors, 300 or 150 anchors, and  $\rho = 0.25$  problems with 5000 sensors, 500 or 250 anchors, and  $\rho = 0.25$

Test problems	$m, m_a, \rho$	$\sigma$	SDP		$B = LL^T$		RMSD		SDPA eTime
			relaxation	Mat. var.	size $L$	size $L$	SDP	w. Grad.	
$m = 3000$	0.0	Sparse	2966	28	18825	525255	4.0e-7	4.0e-7	12.6
$m_a = 300$	0.1	Sparse	2966	28	77369	833934	4.0e-2	1.3e-2	13.1
$\rho = 0.25$	0.2	Sparse	2966	28	77369	833934	6.8e-2	2.6e-2	13.5
$m = 3000$	0.0	Sparse	2955	28	19364	613190	2.6e-6	2.6e-6	18.9
$m_a = 150$	0.1	Sparse	2955	28	69336	873486	6.0e-2	1.5e-2	12.8
$\rho = 0.25$	0.2	Sparse	2955	28	69336	873486	8.4e-2	3.1e-2	13.1
$m = 5000$	0.0	Sparse	4969	25	30683	691716	4.7e-7	4.7e-7	27.8
$m_a = 500$	0.1	Sparse	4969	25	130405	1217155	3.0e-2	1.2e-2	46.8
$\rho = 0.25$	0.2	Sparse	4969	25	130405	1217155	5.4e-2	2.5e-2	40.0
$m = 5000$	0.0	Sparse	4970	26	30810	719808	1.1e-6	1.1e-6	29.3
$m_a = 250$	0.1	Sparse	4970	26	125282	1215716	3.3e-2	1.3e-2	42.4
$\rho = 0.25$	0.2	Sparse	4970	26	125282	1215716	5.6e-2	2.7e-2	46.4



**Fig. 18.5** The depicted solutions and Cholesky factor of SNL problem of 1000 sensors, 100 anchors,  $\rho = 0.25$ , and  $\sigma = 0.2$

## 18.7 Concluding Discussions

We have presented a survey of the sparse SDP relaxation of POPs. The methods described in Sect. 18.3 for exploiting the sparsity characterized by the chordal graph structure were originally proposed for SDP problems [6]. See also [27]. Recently, they have been extended to nonlinear SDP problems in the paper [13], which proposed conversion methods for linear and nonlinear SDP problems into problems with smaller-sized matrix variables and/or smaller-sized matrix inequalities.

Lasserre's dense SDP relaxation was extended to polynomial SDP problems [11, 12, 18], and to more general POPs over symmetric cones [20]. The sparse SDP relaxation in Sect. 18.3 was also extended to polynomial optimization problems over symmetric cones [21]. When we deal with a polynomial SDP problem, we can first apply the conversion methods proposed in [13] to the problem to reduce its size, and then apply the extended sparse SDP relaxation [21] to the reduced polynomial SDP problem. Some numerical results on quadratic SDP problems were shown in [13].

**Acknowledgements** S. Kim's research was supported by NRF 2009-007-1314 and NRF 2010-000-8784. M. Kojima's research was supported by Grant-in-Aid for Scientific Research (B) 22310089.

## References

1. Biswas, P., Liang, T.-C., Toh, K.-C., Wang, T.-C., Ye, Y.: Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineering* **3**, 360–371 (2006)
2. Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network localization. In: Proceedings of the third international symposium on information processing in sensor networks. ACM press, 46–54 (2004)
3. Borchers, B.: SDPLIB 1.2, A library of semidefinite programming test problems. *Optim. Methods Softw.* **11-12**, 683–689 (1999)
4. Choi, M.D., Lam, T.Y., Reznick B.: Sums of squares of real polynomials. *Proc. Symposia Pure Math.* **58**, 103–126 (1995)
5. Conn, A.R., Gould, N.I.M., Toint, P.L.: Testing a class of methods for solving minimization problems with simple bounds on the variables. *Math. Comp.* **50**, 399–430 (1988)
6. Fujisawa, K., Fukuda, M., Kojima, M., Nakata, K.: Numerical evaluation of SDPA (SemiDefinite Programming Algorithm). In: Frenk, H., Roos, K., Terlaky, T., Zhang, S. (eds.) *High Performance Optimization*, pp. 267–301. Kluwer Academic Press (2000)
7. GAMS HomePage <http://www.gams.com/>
8. GLOBAL Library, <http://www.gamsworld.org/global/globallib.htm>
9. Golumbic, M.C.: *Algorithmic graph theory and perfect graphs*. Academic Press, New York (1980)
10. Henrion, D., Lasserre, J.-B.: Detecting global optimality and extracting solutions in GloptiPoly. In: Henrion, D., Garulli, A. (eds.) *Positive Polynomials in Control*. Lecture Notes in Control and Inform. Sci., vol. 312, pp. 293–310. Springer-Verlag, Berlin (2005)
11. Henrion, D., Lasserre, J.-B.: Convergent relaxations of polynomial matrix inequalities and static output feedback. *IEEE Trans. Automatic. Cont.* **51**(2), 192–202 (2006)
12. Hol, C.W.J., Scherer, C.W.: Sum of squares relaxations for polynomial semi-definite programming. In: De Moor, B., Motmans, B. (eds.) *Proceedings of the 16th International Symposium on Mathematical Theory of Networks and Systems*, Leuven, Belgium, 1–10 July 2004
13. Kim, S., Kojima, M., Mevissen, M., Yamashita, M.: Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion. *Math. Program.* **129**(1), 33–68 (2011)
14. Kim, S., Kojima, M., Waki, H.: Generalized Lagrangian duals and sums of squares relaxations of sparse polynomial optimization problems. *SIAM J. Optim.* **15**, 697–719 (2005)
15. Kim, S., Kojima, M., Waki, H.: Exploiting sparsity in SDP relaxation for sensor network localization. *SIAM J. Optim.* **20**(1), 192–215 (2009)
16. Kim, S., Kojima, M., Waki, H., Yamashita, M.: SFSDP: a Sparse version of full semidefinite programming relaxation for sensor network localization problems. To appear in *ACM Trans. Math. Soft. Research Report B-457*, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Meguro, Tokyo 152-8552 (2009)
17. Kobayashi, K., Kim, S., Kojima, M.: Correlative sparsity in primal-dual interior-point methods for LP, SDP and SOCP. *Appl. Math. Optim.*, **58**, 69–88 (2008)
18. Kojima, M.: Sums of squares relaxations of polynomial semidefinite programs. *Research Report B-397*, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Meguro, Tokyo 152-8552 (2003)
19. Kojima, M., Kim, S., Waki, H.: Sparsity in sums of squares of polynomials. *Math. Program.* **103**, 45–62 (2005)
20. Kojima, M., Muramatsu, M.: An extension of sums of squares relaxations to polynomial optimization problems over symmetric cones. *Math. Program.* **110**(2), 315–336 (2007)
21. Kojima, M., Muramatsu, M.: A note on sparse SOS and SDP relaxations for polynomial optimization problems over symmetric cones. *Comp. Optim. Appl.* **42**(1), 31–41 (2009)
22. Lasserre, J.B.: Global optimization with polynomials and the problems of moments. *SIAM J. Optim.* **11**, 796–817 (2001)

23. Lasserre, J.B.: Convergent SDP-relaxations in polynomial optimization with sparsity. *SIAM J. Optim.* **17**(3), 822–843 (2006)
24. Lasserre, J.B.: *Moments, positive polynomials and their applications*. Imperial College Press, London (2010)
25. Lian, T.-C., Wang, T.-C., Ye, Y.: A gradient search method to round the semidefinite programming relaxation solution for ad hoc wireless sensor network localization. Technical report, Dept. of Management Science and Engineering, Stanford University (2004)
26. More, J.J., Garbow, B.S., Hillstrom, K.E.: Testing unconstrained optimization software. *ACM Trans. Math. Soft.* **7**, 17–41 (1981)
27. Nakata, K., Fujisawa, K., Fukuda, M., Kojima, M., Murota, M.: Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results. *Math. Program.* **95**, 303–327 (2003)
28. Putinar, M.: Positive polynomials on compact semi-algebraic sets. *Indiana U. Math. J.* **42**, 969–984 (1993)
29. SDPA Online <http://sdpa.indsys.chuo-u.ac.jp/sdpa/>
30. SFSDP Homepage <http://www.is.titech.ac.jp/~kojima/SFSDP/SFSDP.html>
31. SparsePOP Homepage <http://www.is.titech.ac.jp/~kojima/SparsePOP/>
32. Strum, J.F.: SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Methods Softw.* **11-12**, 625–653 (1999)
33. Tütüncü, R.H., Toh, K.C., Todd, M.J.: Solving semidefinite-quadratic-linear programs using SDPT3. *Math. Program.* **95**, 189–217 (2003)
34. Waki, H., Kim, S., Kojima, M., Muramatsu, M.: Sums of squares and semidefinite programming relaxations for polynomial optimization problems with structured sparsity. *SIAM J. Optim.* **17**, 218–242 (2006)
35. Waki, H., Kim, S., Kojima, M., Muramatsu, M., Sugimoto, H.: Algorithm 883: SparsePOP: A sparse semidefinite programming relaxation of polynomial optimization problems. *ACM Trans. Math. Softw.* **35**(2), 15:1–15:13 (2008)
36. Yamashita, M., Fujisawa, K., Nakata, K., Nakata, M., Fukuda, M., Kobayashi, K., Goto, K.: A high-performance software package for semidefinite programs: SDPA 7. Research Report B-460, Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology, Meguro, Tokyo 152-8552 (2009)

# Chapter 19

## Block Coordinate Descent Methods for Semidefinite Programming

Zaiwen Wen, Donald Goldfarb, and Katya Scheinberg

### 19.1 Introduction

Semidefinite programming (SDP) problems are convex optimization problems that are solvable in polynomial time by interior-point methods [57, 64, 70]. Unfortunately however, in practice large scale SDPs are quite difficult to solve because of the very large amount of work required by each iteration of an interior-point method. Most of these methods form a positive definite  $m \times m$  matrix  $M$ , where  $m$  is the number of linear equality constraints in the standard form SDP, and then compute the search direction by finding the Cholesky factorization of  $M$ . Since  $m$  can be  $O(n^2)$  when the unknown positive semidefinite matrix is  $n \times n$ , it can take  $O(n^6)$  arithmetic operations to do this. Consequently, this becomes impractical both in terms of the time and the amount of memory  $O(m^2)$  required when  $n$  is much larger than one hundred and  $m$  is much larger than a few thousand. Moreover forming  $M$  itself can be prohibitively expensive unless  $m$  is not too large or the constraints in the SDP are very sparse [27]. Although the computational complexities of the block coordinate descent (BCD) methods presented here are not polynomial, each of their iterations can, in certain cases, be executed much more cheaply than in an interior-point algorithm. This enables BCD methods to solve very large instances of these

---

Z. Wen

Department of Mathematics and Institute of Natural Sciences, Shanghai Jiaotong University,  
Shanghai, China

e-mail: [zw2109@sjtu.edu.cn](mailto:zw2109@sjtu.edu.cn)

D. Goldfarb

Department of Industrial Engineering and Operations Research, Columbia University,  
New York, NY, USA

e-mail: [goldfarb@columbia.edu](mailto:goldfarb@columbia.edu)

K. Scheinberg

Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, USA  
e-mail: [kas410@lehigh.edu](mailto:kas410@lehigh.edu)

SDPs efficiently. Preliminary numerical testing verifies this. For example, BCD methods produce highly accurate solutions to maxcut SDP relaxation problems involving matrices of size  $4,000 \times 4,000$  in less than 5.25 minutes and nuclear norm matrix completion SDPs involving matrices of size  $1,000 \times 1,000$  in less than 1 min on a 3.4 GHZ workstation. If only moderately accurate solutions are required (i.e., a relative accuracy of the order of  $10^{-3}$ ) then less than 45 and 10 seconds, respectively, is needed. We note, however, that using a BCD method as a general purpose SDP solver is not a good idea.

### 19.1.1 Review of BCD Methods

BCD methods are among the oldest methods in optimization. Since solving the original problem with respect to all variables simultaneously can be difficult or very time consuming, these approaches are able to reduce the overall computational cost by partitioning the variables into a few blocks and then minimizing the objective function with respect to each block by fixing all other blocks at each inner iteration. They have been studied in convex programming [45, 58], nonlinear programming [6, 33, 34], nonsmooth separable minimization with and without linear constraints [60, 62, 63] and optimization by direct search [42]. Although these methods have never been the main focus of the mathematical optimization community, they remain popular with researchers in the scientific and engineering communities. Recently, interest in coordinate descent methods has been revived due to the wide range of large-scale problems in image reconstruction [9, 24, 73], machine learning including support vector machine training [8, 17, 39, 61], mesh optimization [20], compressive sensing [23, 44, 48, 75], and sparse inverse covariance estimation [2] to which these methods have been successfully applied.

The basic BCD algorithmic strategy can be found under numerous names, including linear and nonlinear Gauss–Seidel methods [34, 35, 53, 69], subspace correction methods [56] and alternating minimization approaches. BCD methods are also closely related to alternating direction augmented Lagrangian (ADAL) methods which alternately minimize the augmented Lagrangian function with respect to different blocks of variables and then update the Lagrange multipliers at each iteration. ADAL methods have been applied to many problem classes, such as, variational inequality problems [36, 37, 72], linear programming [21], nonlinear convex optimization [7, 18, 31, 32, 41, 43, 59], maximal monotone operators [22], nonsmooth  $\ell_1$  minimization arising from compressive sensing [65, 71, 77] and SDP [68, 74].

There are several variants of coordinate and BCD methods. The simplest cyclic (or Gauss–Seidel) strategy is to minimize with respect to each block of variables one after another in a fixed order repeatedly. The essentially cyclic rule [45] selects each block at least once every  $T$  successive iterations, where  $T$  is an integer equal to or greater than the number of blocks. The Gauss–Southwell rule [45, 63] computes a positive value  $q_i$  for every block  $i$  according to some criteria and then chooses the block with the largest value of  $q_i$  to work on next, or chooses the  $k$ -th block to

work on, where  $q_k \geq \beta \max_i q_i$  for  $\beta \in (0, 1]$ . An extreme case is to move along the direction corresponding to the component of the gradient with maximal absolute value [19, 49]. The approach in [49] chooses a block or a coordinate randomly according to pre-specified probabilities for each block or coordinate.

The convergence properties of BCD methods have been intensively studied and we only summarize some results since the 1990s. Bertsekas [6] proved that every limit point generated by the coordinate descent method is a stationary point for the minimization of a general differentiable function  $f(x_1, \dots, x_N)$  over the Cartesian product of closed, nonempty and convex subsets  $\{X_i\}_{i=1}^N$ , such that  $x_i \in X_i$ ,  $i = 1, \dots, N$ , if the minimum of each subproblem is uniquely attained. Grippo and Sciandrone [34] obtained similar results when the objective function  $f$  is componentwise strictly quasiconvex with respect to  $N - 2$  components and when  $f$  is pseudoconvex. Luo and Tseng [45] proved convergence with a linear convergence rate without requiring the objective function to have bounded level sets or to be strictly convex, by considering the problem  $\min_{x \geq 0} g(Ex) + b^\top x$ , where  $g$  is a strictly convex essentially smooth function and  $E$  is a matrix. In [60], Tseng studied nondifferentiable (nonconvex) functions  $f$  with certain separability and regularity properties, and established convergence results when  $f$  is pseudoconvex in every pair of coordinate blocks from among  $N - 1$  coordinate blocks or  $f$  has at most one minimum in each of  $N - 2$  coordinate blocks if  $f$  is continuous on a compact level set, and when  $f$  is quasiconvex and hemivariate in every coordinate block. Tseng and Yun [63] considered a nonsmooth separable problem whose objective function is the sum of a smooth function and a separable convex function, which includes as special cases bound-constrained optimization and smooth optimization with  $\ell_1$ -regularization. They proposed a (block) coordinate gradient descent method with an Armijo line search and established global and linear convergence under a local Lipschitz error bound assumption.

Recently, complexity results for BCD methods have also been explored. Saha and Tewari [54] proved  $O(1/k)$  convergence rates, where  $k$  is the iteration counter, for two cyclic coordinate descent methods for solving  $\min_x f(x) + \lambda \|x\|_1$  under an isotonicity assumption. In [49], Nesterov proposed unconstrained and constrained versions of a Random Coordinate Descent Method (RCDM), and showed that for the class of strongly convex functions, RCDM converges with a linear rate, and showed how to accelerate the unconstrained version of RCDM to have an  $O(1/k^2)$  rate of convergence. A stochastic version of the coordinate descent method with runtime bounds was also considered in [55] for  $\ell_1$ -regularized loss minimization.

### 19.1.2 BCD Methods for SDP

All coordinate descent and block coordinate descent methods for SDP, that maintain positive semidefiniteness of the matrix of variables, are based upon the well known relationship between the positive semidefiniteness of a symmetric matrix and properties of the Schur complement of a sub-matrix of that matrix [76]. We note

that Schur complements play an important role in SDP and related optimization problems. For example, they are often used to formulate problems as SDPs [10, 64]. In [1, 29, 30] they are used to reformulate certain SDP problems as second-order cone programs (SOCPs). More recently, they were used by Banerjee et al. [2] to develop a BCD method for solving the sparse inverse covariance estimation problem whose objective function involves the log determinant of a positive semidefinite matrix. As far as we know, this was the first application of BCD to SDP.

As in the method proposed in Banerjee et. al [2], the basic approach described in this chapter uses Schur complements to develop an *overlapping* BCD method. The coordinates (i.e., variables) in each iteration of these methods correspond to the components of a single row (column) of the unknown semidefinite matrix. Since every row (column) of a symmetric matrix contains one component of each of the other rows (columns), the blocks in these methods overlap. As we shall see below, the convergence result in [6] can be extended to the case of overlapping blocks. However, they do not apply to the case where constraints couple the variables between different blocks. To handle general linear constraints, the BCD methods for SDP described here resort to incorporating these constraints into an augmented Lagrangian function, which is then minimized over each block of variables. Specifically, by fixing any  $(n - 1)$ -dimensional principal submatrix of  $X$  and using its Schur complement, the positive semidefinite constraint is reduced to a simple second-order cone constraint and then a sequence of SOCPs constructed from the primal augmented Lagrangian function are minimized.

Most existing first-order methods for SDP are also based on the augmented Lagrangian method (also referred to as the method of multipliers). Specific methods differ in how the positive semidefinite constraints are handled. In [13, 14], the positive definite variable  $X$  is replaced by  $RR^\top$  in the primal augmented Lagrangian function, where  $R$  is a low rank matrix, and then nonlinear programming approaches are used. In [11, 15], a BCD (alternating minimization) method and an eigenvalue decomposition are used to minimize the primal augmented Lagrangian function. In [78], the positive semidefinite constraint is represented implicitly by using a projection operator and a semismooth Newton approach combined with the conjugate gradient method is applied to minimize the dual augmented Lagrangian function. The regularization methods [47, 50]) and the alternating direction augmented Lagrangian method [68] are also based on a dual augmented Lagrangian approach and the use of an eigenvalue decomposition to maintain complementarity.

We also generalize the BCD approach by using rank-two updates. This strategy also gives rise to SOCP subproblems and enables combinations of the coordinates of the variable matrix  $X$  in more than a single row and column to change at each iteration. Hence, it gives one more freedom in designing an efficient algorithm.

### 19.1.3 Notation and Organization

We adopt the following notation. The sets of  $n \times n$  symmetric matrices and  $n \times n$  symmetric positive semidefinite (positive definite) matrices are denoted by  $\mathcal{S}^n$  and  $\mathcal{S}_+^n$  ( $\mathcal{S}_{++}^n$ ), respectively. The notation  $X \geq 0$  ( $X > 0$ ) is also used to indicate that  $X$  is positive semidefinite (positive definite). Given a matrix  $A \in \mathbb{R}^{n \times n}$ , we denote the  $(i, j)$ -th entry of  $A$  by  $A_{i,j}$ . Let  $\alpha$  and  $\beta$  be given index sets, i.e., subsets of  $\{1, 2, \dots, n\}$ . We denote the cardinality of  $\alpha$  by  $|\alpha|$  and its complement by  $\alpha^c := \{1, 2, \dots, n\} \setminus \alpha$ . Let  $A_{\alpha, \beta}$  denote the submatrix of  $A$  with rows indexed by  $\alpha$  and columns indexed by  $\beta$ , i.e.,

$$A_{\alpha, \beta} := \begin{pmatrix} A_{\alpha_1, \beta_1} & \cdots & A_{\alpha_1, \beta_{|\beta|}} \\ \vdots & & \vdots \\ A_{\alpha_{|\alpha|}, \beta_1} & \cdots & A_{\alpha_{|\alpha|}, \beta_{|\beta|}} \end{pmatrix}.$$

We write  $i$  for the index set  $\{i\}$  and denote the complement of  $\{i\}$  by  $i^c := \{1, 2, \dots, n\} \setminus \{i\}$ . Hence,  $A_{i^c, i^c}$  is the submatrix of  $A$  that remains after removing its  $i$ -th row and column, and  $A_{i^c, i}$  is the  $i$ th column of the matrix  $A$  without the element  $A_{i,i}$ . The inner product between two matrices  $C$  and  $X$  is defined as  $\langle C, X \rangle := \sum_{jk} C_{j,k} X_{j,k}$  and the trace of  $X$  is defined as  $\text{Tr}(X) = \sum_{i=1}^n X_{ii}$ . The vector  $\begin{pmatrix} x \\ y \end{pmatrix}$  obtained by stacking the vector  $x \in \mathbb{R}^p$  on the top of the vector  $y \in \mathbb{R}^q$  is also denoted by  $[x; y] \in \mathbb{R}^{p+q}$ .

The rest of this chapter is organized as follows. In Sect. 19.2, we briefly review the relationship between properties of the Schur complement and the positive semidefiniteness of a matrix, and present a prototype of the RBR method for solving a general SDP. In Sect. 19.3.1, the RBR method is specialized for solving SDPs with only diagonal element constraints, and it is interpreted in terms of the logarithmic barrier function. Coordinate descent methods for sparse inverse covariance estimation proposed in [2] and [26] are reviewed in Sect. 19.3.3. Convergence of the RBR method for SDPs with only simple bound constraints is proved in Sect. 19.3.4. To handle general linear constraints, we apply the RBR method in Sect. 19.4 to a sequence of unconstrained problems using an augmented Lagrangian function approach. Specialized versions for the maxcut SDP relaxation and the minimum nuclear norm matrix completion problem are presented in Sects. 19.4.2 and 19.4.3, respectively. A generalization of the RBR scheme based on a rank-two update is presented in Sect. 19.5. Finally, numerical results for the maxcut and matrix completion problems, are presented in Sect. 19.6 to demonstrate the robustness and efficiency of our algorithms. The methods and numerical results on the RBR methods are collected from the recent technical report [67] and PhD dissertation [66]. The extension to more general rank-two updates is new.

## 19.2 Preliminaries

In this section, we first present a theorem about the Schur complement of a positive (semi-) definite matrix, and then present a RBR prototype method for SDP based on it.

### 19.2.1 Schur Complement

**Theorem 19.1 ([76], Theorems 1.12 and 1.20).** *Let the matrix  $X \in \mathcal{S}^n$  be partitioned as  $X := \begin{pmatrix} \xi & y^\top \\ y & B \end{pmatrix}$ , where  $\xi \in \mathbb{R}$ ,  $y \in \mathbb{R}^{n-1}$  and  $B \in \mathcal{S}^{n-1}$ . The Schur complement of  $B$  in  $X$  is defined as  $(X/B) := \xi - y^\top B^\dagger y$ , where  $B^\dagger$  is the Moore–Penrose pseudo-inverse of  $B$ . Then the following holds.*

- (1) *If  $B$  is nonsingular, then  $X > 0$  if and only if  $B > 0$  and  $(X/B) > 0$ .*
- (2) *If  $B$  is nonsingular, then  $X \geq 0$  if and only if  $B > 0$  and  $(X/B) \geq 0$ .*
- (3)  *$X \geq 0$  if and only if  $B \geq 0$ ,  $(X/B) \geq 0$  and  $y \in \mathcal{R}(B)$ , where  $\mathcal{R}(B)$  is the range space of  $B$ .*

*Proof.* We only prove here (1) and (2). Since  $B$  is nonsingular,  $X$  can be factorized as

$$X = \begin{pmatrix} 1 & y^\top B^{-1} \\ 0 & I \end{pmatrix} \begin{pmatrix} \xi - y^\top B^{-1}y & 0 \\ 0 & B \end{pmatrix} \begin{pmatrix} 1 & 0 \\ B^{-1}y & I \end{pmatrix}. \quad (19.1)$$

Hence,  $\det(X) = (\xi - y^\top B^{-1}y)\det(B)$  and

$$X > (\geq) 0 \iff B > 0 \text{ and } (X/B) := \xi - y^\top B^{-1}y > (\geq) 0. \quad (19.2)$$

□

### 19.2.2 A RBR Method Prototype for SDP

We consider here the standard form SDP problem

$$\begin{aligned} \min_{X \in \mathcal{S}^n} \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \mathcal{A}(X) = b, \quad X \geq 0, \end{aligned} \quad (19.3)$$

where the linear map  $\mathcal{A}(\cdot) : \mathcal{S}^n \rightarrow \mathbb{R}^m$  is defined by

$$\mathcal{A}(X) := (\langle A^{(1)}, X \rangle, \dots, \langle A^{(m)}, X \rangle)^\top,$$

the matrices  $C, A^{(i)} \in \mathcal{S}^n$ , and the vector  $b \equiv (b_1, \dots, b_m)^\top \in \mathbb{R}^m$  are given. Henceforth, the following Slater condition for (19.3) is assumed to hold.

**Assumption 19.1** Problem (19.3) satisfies the Slater condition:

$$\begin{cases} \mathcal{A}: \mathcal{S}^n \rightarrow \mathbb{R}^m \text{ is onto ,} \\ \exists X^1 \in \mathcal{S}_{++}^n \text{ such that } \mathcal{A}(X^1) = b . \end{cases} \quad (19.4)$$

Given a strictly feasible solution  $X^k > 0$ , we can construct a SOCP restriction for the SDP problem (19.3) as follows. Fix the  $n(n-1)/2$  variables in the  $(n-1) \times (n-1)$  submatrix  $B := X_{1^c, 1^c}^k$  of  $X^k$  and let  $\xi$  and  $y$  denote the remaining unknown variables  $X_{1,1}$  and  $X_{1^c, 1}$  (i.e., row 1/column 1), respectively. Hence, the matrix  $X := \begin{pmatrix} \xi & y^\top \\ y & B \end{pmatrix} := \begin{pmatrix} \xi & y^\top \\ y & X_{1^c, 1^c}^k \end{pmatrix}$ . It then follows from Theorem 19.1 that  $X \geq 0$  is equivalent to  $\xi - y^\top B^{-1}y \geq 0$ . Here we write this as  $\xi - y^\top B^{-1}y \geq \nu$ , with  $\nu = 0$ , so that strict positive definiteness of  $X$  can be maintained if we choose  $\nu > 0$ . Hence, the SDP problem (19.3) becomes

$$\begin{aligned} \min_{[\xi; y] \in \mathbb{R}^n} \quad & \bar{c}^\top [\xi; y] \\ \text{s.t.} \quad & \bar{A} [\xi; y] = \bar{b}, \\ & \xi - y^\top B^{-1}y \geq \nu, \end{aligned} \quad (19.5)$$

where  $\nu = 0$ , and  $\bar{c}$ ,  $\bar{A}$  and  $\bar{b}$  are defined as follows using the subscript  $i = 1$ :

$$\bar{c} := \begin{pmatrix} C_{i,i} \\ 2C_{i^c, i} \end{pmatrix}, \quad \bar{A} := \begin{pmatrix} A_{i,i}^{(1)} & 2A_{i,i^c}^{(1)} \\ \dots & \dots \\ A_{i,i}^{(m)} & 2A_{i,i^c}^{(m)} \end{pmatrix} \quad \text{and} \quad \bar{b} := \begin{pmatrix} b_1 - \langle A_{i^c, i}^{(1)}, B \rangle \\ \dots \\ b_m - \langle A_{i^c, i}^{(m)}, B \rangle \end{pmatrix}. \quad (19.6)$$

If we let  $LL^\top = B$  be the Cholesky factorization of  $B$  and introduce a new variable  $z = L^{-1}y$ , the Schur complement constraint  $\xi - y^\top B^{-1}y \geq \nu$  is equivalent to the linear constraints  $Lz = y$  and  $\eta = \xi - \nu$  and the rotated second-order cone constraint  $\|z\|_2^2 \leq \eta$ . Clearly, similar problems can be constructed if for any  $i$ ,  $i = 1, \dots, n$ , all elements of  $X^k$  other than those in the  $i$ -th row/column are fixed and only the elements in the  $i$ -th row/column are treated as unknowns.

We now present the RBR method for solving (19.3). Starting from a positive definite feasible solution  $X^1$ , we update one row/column of the solution  $X$  at each of  $n$  inner steps by solving subproblems of the form (19.5) with  $\nu > 0$ . As we shall show below, choosing  $\nu > 0$  in (19.5) (i.e., keeping all iterates positive definite), is necessary for the RBR method to be well-defined. This procedure from the first row to the  $n$ -th row is called a *cycle*. At the first step of the  $k$ -th cycle, we fix  $B := X_{1^c, 1^c}^k$ , and solve subproblem (19.5), whose solution is denoted by  $[\xi; y]$ . Then the first row/column of  $X^k$  is replaced by  $X_{1,1}^k := \xi$  and  $X_{1^c, 1}^k := y$ . Similarly, we set  $B := X_{i^c, i^c}^k$  in the  $i$ -th inner iteration and assign the parameters  $\bar{c}$ ,  $\bar{A}$  and  $\bar{b}$  according to (19.6). Then the solution  $[\xi; y]$  of (19.5) is used to set  $X_{i,i}^k := \xi$  and  $X_{i^c,i}^k := y$ . The  $k$ -th cycle is finished after the  $n$ -th row/column is updated. Then we set  $X^{k+1} := X^k$  and repeat this procedure until the relative decrease in the objective function on a cycle

**Algorithm 5** A RBR method prototype

---

Set  $X^1 > 0$ ,  $\nu \geq 0$ ,  $k := 1$  and  $\epsilon \geq 0$ . Set  $F^0 := +\infty$  and compute  $F^1 := \langle C, X^1 \rangle$ .

**while**  $\frac{F^{k-1} - F^k}{\max\{|F^{k-1}|, 1\}} \geq \epsilon$  **do**

**for**  $i = 1, \dots, n$  **do**

Set  $B := X_{i,i}^k$  and the parameters  $\tilde{c}$ ,  $\tilde{A}$  and  $\tilde{b}$  according to (19.6).

Solve the subproblem (19.5) whose solution is denoted by  $\xi$  and  $y$ .

Update  $X_{i,i}^k := \xi$ ,  $X_{i,i}^k := y$  and  $X_{i,i}^k := y^\top$ .

Compute  $F^k := \langle C, X^k \rangle$ . Set  $X^{k+1} := X^k$  and  $k := k + 1$ .

---

becomes smaller than some tolerance  $\epsilon$ . This RBR method prototype is outlined in Algorithm 5. In the next section, we illustrate its usefulness for problems in which the linear constraints are simple bound constraints. Unfortunately, when they are not, the RBR prototype fails. Hence, for the general case we present an augmented Lagrangian version of the RBR method in Sect. 19.4.

We note that the RBR method is similar to the block *Gauss–Seidel* method for solving a system of linear equations and the block coordinate descent method (sometimes referred to as the nonlinear Gauss–Seidel method) for nonlinear programming, except that because of the symmetry of  $X$ , the blocks in the RBR method overlap. Specifically, exactly one of the variables in any two inner iterations of the RBR method overlap.

## 19.3 The RBR Methods for SDPs with Bound Constraints

We now apply the RBR method to SDPs with simple bound constraints, including the maxcut SDP relaxation, the SDP relaxation of the matrix completion problem, and the sparse inverse covariance estimation problem. Convergence of the RBR method for such problems is also analyzed.

### 19.3.1 Maxcut SDP Relaxation

The well known SDP relaxation [3, 12, 28, 38] for the maxcut problem, which seeks to partition the vertices of a graph into two sets so that the sum of the weighted edges connecting vertices in one set with vertices in the other set is maximized, takes the following form:

$$\begin{aligned} \min_{X \geq 0} \quad & \langle C, X \rangle \\ \text{s.t.} \quad & X_{ii} = 1, \quad i = 1, \dots, n. \end{aligned} \tag{19.7}$$

We now present the RBR subproblem for solving (19.7). Since the diagonal elements of  $X$  are known to be equal to 1, they are kept fixed at 1. At the  $i$ th step of the  $k$ -th cycle, we fix  $B = X_{j^*, j^*}^k$ , where  $X^k$  is the iterate at the  $(i-1)$ -st step of the  $k$ -th cycle. Although in all RBR algorithms positive definiteness of all iterates is maintained, we assume here that  $B$  is positive semidefinite and use the generalized Schur complement to construct the second-order cone constraint. Hence, the RBR subproblem (19.5) for problem (19.7) is

$$\begin{aligned} \min_{y \in \mathbb{R}^{n-1}} & \quad \widehat{c}^\top y \\ \text{s.t.} & \quad 1 - y^\top B^\dagger y \geq \nu, \quad y \in \mathcal{R}(B), \end{aligned} \quad (19.8)$$

where  $\widehat{c} := 2C_{j^*, i}$ .

**Lemma 19.1.** *If  $\gamma := \widehat{c}^\top B \widehat{c} > 0$ , the solution of problem (19.8) with  $\nu < 1$  is given by*

$$y = -\sqrt{\frac{1-\nu}{\gamma}} B \widehat{c}. \quad (19.9)$$

Otherwise,  $y = 0$  is a solution.

*Proof.* Suppose that the matrix  $B \in \mathcal{S}_+^n$  has rank  $r$ , where  $0 < r \leq n$ . Hence,  $B$  has the spectral decomposition

$$B = Q \Lambda Q^\top = (Q_r \ Q_l) \begin{pmatrix} \Lambda_r & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} Q_r^\top \\ Q_l^\top \end{pmatrix} = Q_r \Lambda_r Q_r^\top. \quad (19.10)$$

where  $Q$  is an orthogonal matrix,  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_r, 0, \dots, 0)$ , and  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_r > 0$ , and the Moore–Penrose pseudo-inverse of  $B$  is

$$B^\dagger = (Q_r \ Q_l) \begin{pmatrix} \Lambda_r^{-1} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} Q_r^\top \\ Q_l^\top \end{pmatrix} = Q_r \Lambda_r^{-1} Q_r^\top.$$

Let  $z = Q^\top y =: [z_r; z_l]$ . Since  $y \in \mathcal{R}(B)$  and  $\mathcal{R}(B) = \mathcal{R}(Q_r)$ ,  $z_l = 0$ ; hence, problem (19.8) is equivalent to

$$\begin{aligned} \min_{z_r \in \mathbb{R}^r} & \quad (Q_r^\top \widehat{c})^\top z_r \\ \text{s.t.} & \quad 1 - z_r^\top \Lambda_r^{-1} z_r \geq \nu, \end{aligned} \quad (19.11)$$

whose Lagrangian function is  $\ell(z_r, \lambda) = (Q_r^\top \widehat{c})^\top z_r - \frac{\lambda}{2}(1 - \nu - z_r^\top \Lambda_r^{-1} z_r)$ , where  $\lambda \geq 0$ . At an optimal solution  $z_r^*$  to (19.11),

$$\nabla_{z_r} \ell(z_r^*, \lambda^*) = Q_r^\top \widehat{c} + \lambda^* \Lambda_r^{-1} z_r^* = 0. \quad (19.12)$$

Suppose that  $1 - (z_r^*)^\top \Lambda^{-1} z_r^* > \nu$ . It follows from the complementary conditions that  $\lambda^* = 0$ , which implies that  $Q_r^\top \widehat{c} = 0$  and  $\gamma = 0$  by using (19.12). It is obvious that  $y^* = 0$  is a solution. Otherwise,  $z_r^*$  satisfies the constraint (19.11) with equality, i.e.,  $1 - (z_r^*)^\top \Lambda^{-1} z_r^* = \nu$ . Then, we have  $z_r^* = -\Lambda_r Q_r^\top \widehat{c} / \lambda^*$  and

$$1 - \frac{\widehat{c}^\top Q_r \Lambda_r \Lambda_r^{-1} \Lambda_r Q_r^\top \widehat{c}}{(\lambda^*)^2} = 1 - \frac{\gamma}{(\lambda^*)^2} = \nu.$$

Since  $\nu < 1$ , we must have  $\gamma > 0$ . Hence, we obtain  $\lambda^* = \sqrt{\gamma/(1-\nu)}$  and

$$y^* = Q_r z_r^* = -\sqrt{\frac{1-\nu}{\gamma}} Q_r \Lambda_r Q_r^\top \widehat{c} = -\sqrt{\frac{1-\nu}{\gamma}} B \widehat{c}.$$

□

For simplicity, we let PURE-RBR-M denote the RBR method for the maxcut SDP described above. PURE-RBR-M is extremely simple since only a single matrix-vector product is involved at each inner step. Numerical experiments show PURE-RBR-M works fine if the initial solution  $X$  is taken as the identity matrix even if we take  $\nu = 0$ . However, there exist examples where starting from a rank-one point that is not optimal, the RBR method using  $\nu = 0$  either does not move away from the initial solution or it moves to a non-optimal rank-one solution and stays there.

We next interpret PURE-RBR-M as a variant of the RBR method applied to a logarithmic barrier function approximation to (19.7). Consider the logarithmic barrier problem for (19.7), i.e.,

$$\begin{aligned} \min_{X \in \mathcal{S}^n} \quad & \phi_\sigma(X) := \langle C, X \rangle - \sigma \log \det X \\ \text{s.t.} \quad & X_{ii} = 1, \forall i = 1, \dots, n, \quad X > 0, \end{aligned} \tag{19.13}$$

where we define  $\log \det(X)$  to be negative infinity for  $X$  not positive definite. Given a row  $i$  and fixing the block  $B = X_{i^c, i^c}$ , we have from (19.1) that

$$\det(X) = \det(B)(1 - X_{i^c, i}^\top B^{-1} X_{i^c, i}),$$

which implies that

$$\phi_\sigma(X) := \widehat{c}^\top X_{i^c, i} - \sigma \log(1 - X_{i^c, i}^\top B^{-1} X_{i^c, i}) + w(B),$$

where  $\widehat{c} = 2C_{i^c, i}$  and  $w(B)$  is a function of  $B$  (i.e., a constant). Hence, the RBR subproblem for (19.13) is the unconstrained minimization problem

$$\min_{y \in \mathbb{R}^{n-1}} \quad \widehat{c}^\top y - \sigma \log(1 - y^\top B^{-1} y). \tag{19.14}$$

Lemma 19.2 below shows that PURE-RBR-M is essentially the RBR method applied to solving problem (19.13), if in the former algorithm  $\nu$  is replaced by  $2\sigma \frac{\sqrt{\sigma^2 + \gamma} - \sigma}{\gamma}$ . If  $B$  is only positive semidefinite the (19.14) is replaced by

$$\min_{y \in \mathbb{R}^{n-1}} \quad \widehat{c}^\top y - \sigma \log(1 - y^\top B^\dagger y), \quad \text{s.t. } y \in \mathcal{R}(B). \quad (19.15)$$

**Lemma 19.2.** *If  $\gamma := \widehat{c}^\top B \widehat{c} > 0$ , the solution of problem (19.15) is*

$$y = -\frac{\sqrt{\sigma^2 + \gamma} - \sigma}{\gamma} B \widehat{c}. \quad (19.16)$$

Hence, the subproblem (19.8) has the same solution as (19.15) if  $\nu = 2\sigma \frac{\sqrt{\sigma^2 + \gamma} - \sigma}{\gamma}$ .

*Proof.* Similar to Lemma 19.1, we have the spectral decomposition (19.10) of  $B$ . Let  $z = Q^\top y =: [z_r; z_l]$ . Since  $y \in \mathcal{R}(B)$  and  $\mathcal{R}(B) = \mathcal{R}(Q_r)$ , we obtain  $z_l = 0$  and hence  $y = Q_r z_r$ . Therefore, problem (19.15) is equivalent to

$$\min_{z_r} \quad (Q_r^\top \widehat{c})^\top z_r - \sigma \log(1 - z_r^\top \Lambda_r^{-1} z_r), \quad (19.17)$$

whose first-order optimality conditions are

$$Q_r^\top \widehat{c} + \frac{2\sigma \Lambda_r^{-1} z_r^*}{1 - (z_r^*)^\top \Lambda_r^{-1} z_r^*} = 0, \text{ and } 1 - (z_r^*)^\top \Lambda_r^{-1} z_r^* > 0. \quad (19.18)$$

Let  $\theta = 1 - (z_r^*)^\top \Lambda_r^{-1} z_r^*$ . Then (19.18) implies that  $z_r^* = -\frac{\theta \Lambda_r Q_r^\top \widehat{c}}{2\sigma}$ . Substituting this expression for  $z_r^*$  into the definition of  $\theta$ , we obtain  $\theta^2 \frac{\gamma}{4\sigma^2} + \theta - 1 = 0$ , which has a positive root  $\theta = \frac{2\sigma \sqrt{\sigma^2 + \gamma} - 2\sigma^2}{\gamma}$ . Hence,  $y^* = -\frac{\sqrt{\sigma^2 + \gamma} - \sigma}{\gamma} B \widehat{c}$ . Since  $\nabla^2 \phi_\sigma(y) \succeq 0$ ,  $y^*$  is an optimal solution of (19.15). Furthermore, problems (19.8) and (19.15) are equivalent if  $\frac{\sqrt{\sigma^2 + \gamma} - \sigma}{\gamma} = \sqrt{\frac{1-\nu}{\nu}}$ ; that is  $\nu = 2\sigma \frac{\sqrt{\sigma^2 + \gamma} - \sigma}{\gamma}$ .  $\square$

*Remark 19.1.* Note from (19.16) that  $\lim_{\sigma \rightarrow 0} y = -\frac{B \widehat{c}}{\sqrt{\gamma}}$ .

### 19.3.2 Matrix Completion

Given a matrix  $M \in \mathbb{R}^{p \times q}$  and an index set

$$\mathcal{Q} \subseteq \{(i, j) \mid i \in \{1, \dots, p\}, j \in \{1, \dots, q\}\},$$

the nuclear norm matrix completion problem is

$$\begin{aligned} \min_{W \in \mathbb{R}^{p \times q}} \quad & \|W\|_* \\ \text{s.t.} \quad & W_{ij} = M_{ij}, \forall (i, j) \in \mathcal{Q}. \end{aligned} \quad (19.19)$$

An equivalent SDP formulation of (19.19) is

$$\begin{aligned} & \min_{X \in \mathcal{S}^n} \text{Tr}(X) \\ \text{s.t.} \quad & X := \begin{bmatrix} X^{(1)} & W \\ W^\top & X^{(2)} \end{bmatrix} \geq 0 \\ & W_{ij} = M_{ij}, \forall (i, j) \in \Omega, \end{aligned} \quad (19.20)$$

where  $n = p + q$  and the number of linear constraints is  $m = |\Omega|$ . Let  $M_\Omega$  be the vector whose elements are the components of  $\{M_{i,j} \mid (i, j) \in \Omega\}$  obtained by stacking the columns of  $M$  from column 1 to column  $q$  and then keeping only those elements that are in  $\Omega$ . Hence,  $M_\Omega$  corresponds to the right hand side  $b$  of the constraints in the general SDP (19.3).

We now present the RBR subproblem (19.46) corresponding to problem (19.20). First, the vector  $y$  can be partitioned into two subvectors corresponding to elements whose indices are, respectively, in and not in the set  $\Omega$ :

$$y := \begin{pmatrix} \widehat{y} \\ \widetilde{y} \end{pmatrix}, \quad \widehat{y} := X_{\alpha,i}, \text{ and } \widetilde{y} := X_{\beta,i},$$

where, the index sets  $\beta := i^c \setminus \alpha$  and

$$\alpha := \begin{cases} \{j+p, \mid j \in \bar{\alpha}\}, \text{ where } \bar{\alpha} := \{j \mid (i, j) \in \Omega, j = 1, \dots, q\}, \text{ if } i \leq p, \\ \{j \mid (j, i) \in \Omega, j = 1, \dots, p\}, \text{ if } p < i \leq n. \end{cases} \quad (19.21)$$

Letting

$$\widetilde{b} := \begin{cases} (M_{i,\bar{\alpha}})^\top, & \text{if } i \leq p, \\ M_{\alpha,i-p}, & \text{if } p < i \leq n, \end{cases} \quad (19.22)$$

the RBR subproblem (19.5) becomes

$$\begin{aligned} & \min_{(\xi; y) \in \mathbb{R}^n} \quad \xi \\ \text{s.t.} \quad & \widehat{y} = \widetilde{b}, \quad \xi - y^\top B^{-1} y \geq \nu, \end{aligned} \quad (19.23)$$

where the matrix  $B = \begin{pmatrix} X_{\alpha,\alpha}^k & X_{\alpha,\beta}^k \\ X_{\beta,\alpha}^k & X_{\beta,\beta}^k \end{pmatrix}$ .

**Lemma 19.3.** *The optimal solution of the RBR subproblem (19.23) is given by*

$$\xi = \lambda^\top \widetilde{b} + \nu, \quad \widehat{y} = X_{\beta,\alpha}^k \lambda, \quad \text{where, } \lambda = (X_{\alpha,\alpha}^k)^{-1} \widetilde{b}. \quad (19.24)$$

*Proof.* Note that the optimal solution  $[\xi; y] = [\xi; \widehat{y}; \widetilde{y}]$  of (19.23) must satisfy  $\xi = y^\top B^{-1} y + v$ . Hence, (19.23) is equivalent to the linearly constrained quadratic minimization problem

$$\min_y \{y^\top B^{-1} y \mid \widehat{y} = \widetilde{y}\} \quad (19.25)$$

whose optimality conditions are

$$\begin{pmatrix} X_{\alpha,\alpha}^k & X_{\alpha,\beta}^k \\ X_{\beta,\alpha}^k & X_{\beta,\beta}^k \end{pmatrix}^{-1} \begin{pmatrix} \widehat{y} \\ \widetilde{y} \end{pmatrix} - \begin{pmatrix} \lambda \\ \mathbf{0} \end{pmatrix} = 0, \quad (19.26)$$

which implies that

$$\begin{pmatrix} \widehat{y} \\ \widetilde{y} \end{pmatrix} = \begin{pmatrix} \widetilde{b} \\ \widetilde{y} \end{pmatrix} = \begin{pmatrix} X_{\alpha,\alpha}^k \\ X_{\beta,\alpha}^k \end{pmatrix} \lambda.$$

□

Note from (19.24) that we only need to solve a single system of linear equations, whose size is the number of known elements in the row and hence expected to be small, to obtain the minimizer of the RBR subproblem (19.23).

### 19.3.3 Sparse Inverse Covariance Estimation

In this subsection, we review the block coordinate descent methods proposed in [2] and [26] for solving the sparse inverse covariance estimation problem. Given an empirical covariance matrix  $S \in \mathcal{S}^n$ , the problem is to maximize the  $\ell_1$ -penalized log-likelihood function, i.e.,

$$\hat{\Sigma}^{-1} = \arg \max_{X > 0} \log \det X - \text{Tr}(S X) - \lambda \|X\|_1, \quad (19.27)$$

where  $\lambda > 0$  and  $\|X\|_1 = \sum_{i,j} |X_{i,j}|$ . Instead of solving (19.27) directly, the approaches in [2] and [26] consider the dual of (19.27)

$$\hat{\Sigma} = \arg \max_{W > 0} \log \det W, \quad \text{s.t.} \quad \|W - S\|_\infty \leq \lambda, \quad (19.28)$$

which is a problem with only simple bound constraints. To derive this, note that (19.27) is equivalent to

$$\max_{X > 0} \min_{\|U\|_\infty \leq \lambda} \log \det X - \text{Tr}(X(S + U)), \quad (19.29)$$

since the  $\ell_1$ -norm  $\|X\|_1$  can be expressed as  $\max_{\|U\|_\infty \leq 1} \text{Tr}(XU)$ , where  $\|U\|_\infty$  is the maximum of the absolute values of the elements of the symmetric matrix  $U$ . It is obvious that

$$-\log \det(S + U) - n = \max_{X > 0} \log \det X - \text{Tr}(X(S + U)).$$

Hence, the dual (19.28) is obtained by exchanging the max and the min in (19.29).

The subproblems solved at each iteration of the BCD methods in [2] and [26] are constructed as follows. Given a positive definite matrix  $W > 0$ ,  $W$  and  $S$  are partitioned according to the same pattern as

$$W = \begin{pmatrix} \xi & y^\top \\ y & B \end{pmatrix} \text{ and } S = \begin{pmatrix} \xi_S & y_S^\top \\ y_S & B_S \end{pmatrix},$$

where  $\xi, \xi_S \in \mathbb{R}$ ,  $y, y_S \in \mathbb{R}^{n-1}$  and  $B, B_S \in \mathbb{S}^{n-1}$ . Since  $\log \det W = \log(\xi - y^\top B^{-1} y) \det B$ , and  $B$  is fixed, the RBR subproblem for (19.28) becomes the quadratic program

$$\min_{[\xi;y]} y^\top B^{-1} y - \xi, \quad \text{s.t.} \quad \|[\xi;y] - [\xi_S;y_S]\|_\infty \leq \lambda, \quad \xi \geq 0. \quad (19.30)$$

Note that (19.30) is separable in  $y$  and  $\xi$ . The solution  $\xi$  is equal to  $\xi_S + \lambda$ . In fact, the first-order optimality conditions of (19.27) and  $X > 0$  imply that  $W_{ii} = S_{ii} + \lambda$  for  $i = 1, \dots, n$ . Hence, problem (19.30) reduces to

$$\min_y y^\top B^{-1} y, \quad \text{s.t.} \quad \|y - y_S\|_\infty \leq \lambda. \quad (19.31)$$

It can be verified that the dual of (19.31) is

$$\min_x x^\top B x - y_S^\top x + \lambda \|x\|_1, \quad (19.32)$$

which is also equivalent to

$$\min_x \left\| B^{\frac{1}{2}} x - \frac{1}{2} B^{-\frac{1}{2}} y_S \right\|_2^2 + \lambda \|x\|_1. \quad (19.33)$$

If  $x$  solves (19.33), then  $y = Bx$  solves (19.31).

The BCD method in [2] solves a sequence of constrained problems (19.31). Specifically, the initial point is set to  $W^1 = S + \lambda I$  so that only off-diagonal elements have to be updated. The parameters  $B := W_{i^c, i^c}^k$ ,  $y_S = S_{i^c, i}$  and  $B_S = S_{i^c, i^c}$  are assigned in the  $i$ -th inner iteration at  $k$ -th cycle. Then the solution  $y$  of (19.30) is computed and one sets  $W_{i^c, i}^k := y$ . A similar procedure is used in the approach in [26] except that the solution  $y$  is obtained by solving the so-called LASSO problem (19.33) using a coordinate descent algorithm, which does not require computation of either  $B^{\frac{1}{2}}$  or  $B^{-\frac{1}{2}}$ .

### 19.3.4 Convergence Results

The RBR method can be extended to solve

$$\begin{aligned} \min_{X \in \mathcal{S}^n} \quad & \psi_\sigma(X) := f(X) - \sigma \log \det X \\ \text{s.t.} \quad & X \in \mathcal{X} := \{X \in \mathcal{S}^n \mid L \leq X \leq U, X > 0\}. \end{aligned} \quad (19.34)$$

where  $f(X)$  is a differentiable convex function of  $X$ , the constant matrices  $L, U \in \mathcal{S}^n$  satisfy  $L \leq U$  and  $L \leq X$  means that  $L_{i,j} \leq X_{i,j}$  for all  $i, j = 1, \dots, n$ . Note that  $L_{i,j} = -\infty$  ( $U_{i,j} = \infty$ ) if  $X_{i,j}$  is unbounded below (above). Clearly, problem (19.34) includes (19.13) and the logarithmic barrier function version of problems (19.20) and (19.28) as special cases. Starting from the point  $X^k > 0$  at the  $k$ -th cycle, we fix the  $n(n-1)/2$  variables in the  $(n-1) \times (n-1)$  submatrix  $B := X_{i^c, i^c}^k$  of  $X^k$  and let  $\xi$  and  $y$  denote the remaining unknown variables  $X_{i,i}$  and  $X_{i^c, i}$  (i.e., row  $i$ /column  $i$ ), respectively; i.e.,  $X^k \approx \begin{pmatrix} \xi & y^\top \\ y & B \end{pmatrix}$ . Hence, the RBR subproblem for problem (19.34) becomes

$$\begin{aligned} \min_{X \in \mathcal{S}^n} \quad & \tilde{f}(\xi, y) - \sigma \log(\xi - y^\top B^{-1} y) \\ \text{s.t.} \quad & \begin{pmatrix} L_{i,i} \\ L_{i^c, i} \end{pmatrix} \leq \begin{pmatrix} \xi \\ y \end{pmatrix} \leq \begin{pmatrix} U_{i,i} \\ U_{i^c, i} \end{pmatrix}, \end{aligned} \quad (19.35)$$

where  $\tilde{f}(\xi, y) := f(X^k)$ . Inspired by Proposition 2.7.1 in [6], we now prove the following convergence result for the RBR method applied to problem (19.34).

**Theorem 19.2.** *Let  $\{X^k\}$  be a sequence generated by the RBR method for solving (19.34). Assume that the level set  $\{X \in \mathcal{X} | \psi_\sigma(X) \leq \psi_\sigma(X^1)\}$  is compact. Then every limit point of  $\{X^k\}$  is a global minimizer of (19.34).*

*Proof.* Clearly, the RBR method produces a sequence of nondecreasing objective function values

$$\psi_\sigma(X^k) \geq \psi_\sigma(X^{k,1}) \geq \psi_\sigma(X^{k,2}) \geq \dots \geq \psi_\sigma(X^{k,n-1}) \geq \psi_\sigma(X^{k+1}). \quad (19.36)$$

Let  $\tilde{X}$  be a limit point of the sequence  $\{X^k\}$ . It follows from (19.36) that the sequences  $\{\psi_\sigma(\tilde{X}^k)\}$ ,  $\{\psi_\sigma(\tilde{X}^{k,1})\}$ ,  $\dots$ ,  $\{\psi_\sigma(\tilde{X}^{k,n-1})\}$  all converge to a bounded number  $\psi_\sigma(\tilde{X})$ . Hence,  $\tilde{X}$  must be positive definite. We now show that  $\tilde{X}$  minimizes  $\psi_\sigma(X)$ .

Let  $\{X^{k_j}\}$  be a subsequence of  $\{X^k\}$  that converges to  $\tilde{X}$ . We first show that  $\{X^{k_j,1} - X^{k_j}\}$  converges to zero as  $j \rightarrow \infty$ . Assume on the contrary, that  $\{X^{k_j,1} - X^{k_j}\}$  does not converge to zero. Then there exists a subsequence  $\{\bar{k}_j\}$  of  $\{k_j\}$  and some  $\bar{\gamma} > 0$  such that  $\gamma^{\bar{k}_j} := \|X^{\bar{k}_j,1} - X^{\bar{k}_j}\|_F \geq \bar{\gamma}$  for all  $j$ . Let  $D^{\bar{k}_j,1} := (X^{\bar{k}_j,1} - X^{\bar{k}_j})/\gamma^{\bar{k}_j}$ . Thus  $X^{\bar{k}_j,1} = X^{\bar{k}_j} + \gamma^{\bar{k}_j} D^{\bar{k}_j,1}$ ,  $\|D^{\bar{k}_j,1}\|_F = 1$  and  $D^{\bar{k}_j,1}$  differs from zero only along the first row/column. Since  $D^{\bar{k}_j,1}$  belongs to a compact set, it has a limit point  $\bar{D}^1$ . Hence, there exists a subsequence of  $\{\hat{k}_j\}$  of  $\{\bar{k}_j\}$  such that  $D^{\hat{k}_j,1}$  converges to  $\bar{D}^1$ . Consider an arbitrary  $t \in [0, 1]$ . Since  $0 \leq t\bar{\gamma} \leq \gamma^{\hat{k}_j}$ ,  $X^{\hat{k}_j} + tD^{\hat{k}_j,1}$  lies on the segment joining  $X^{\hat{k}_j}$  and  $X^{\hat{k}_j} + \gamma^{\hat{k}_j} D^{\hat{k}_j,1} = X^{\hat{k}_j,1}$ , and belongs to  $\mathcal{X}$  since  $\mathcal{X}$  is a convex set. Moreover, since  $X^{\hat{k}_j,1}$  uniquely minimizes  $\psi_\sigma(X)$  over all  $X$  that differ from  $X^{\hat{k}_j}$  along the first row/column, it follows from the convexity of  $\psi_\sigma(X)$  that

$$\psi_\sigma(X^{\hat{k}_j,1}) = \psi_\sigma(X^{\hat{k}_j} + \gamma^{\hat{k}_j} D^{\hat{k}_j,1}) \leq \psi_\sigma(X^{\hat{k}_j} + t\gamma^{\hat{k}_j} D^{\hat{k}_j,1}) \leq \psi_\sigma(X^{\hat{k}_j}). \quad (19.37)$$

Since  $\psi_\sigma(X^{k_j,1})$  converges to  $\psi_\sigma(\tilde{X})$ , it follows (19.37) that  $\psi_\sigma(\tilde{X}) \leq \psi_\sigma(\tilde{X} + t\bar{\gamma}\bar{D}^1) \leq \psi_\sigma(\tilde{X})$ , which implies that  $\psi_\sigma(\tilde{X}) = \psi_\sigma(\tilde{X} + t\bar{\gamma}\bar{D}^1)$  for all  $t \in [0, 1]$ . Since  $\bar{\gamma}\bar{D}^1 \neq 0$ , this contradicts the fact that  $\psi_\sigma(X)$  is strictly convex; hence  $X^{k_j,1} - X^{k_j}$  converges to zero and  $X^{k_j,1}$  converges to  $\tilde{X}$ .

From the definition (19.34), we have  $\psi_\sigma(X^{k_j,1}) \leq \psi_\sigma(X)$  for all

$$X \in V^{k_j,1} := \left\{ \begin{pmatrix} \xi & y^\top \\ y & X_{1^c,1^c}^{k_j} \end{pmatrix} \middle| \begin{pmatrix} \xi \\ y \end{pmatrix} \in \mathbb{R}^n, \begin{pmatrix} L_{1,1} \\ L_{1^c,1} \end{pmatrix} \leq \begin{pmatrix} \xi \\ y \end{pmatrix} \leq \begin{pmatrix} U_{1,1} \\ U_{1^c,1} \end{pmatrix} \right\}.$$

Taking the limit as  $j$  tends to infinity, we obtain that  $\psi_\sigma(\tilde{X}) \leq \psi_\sigma(X)$  for all

$$X \in V^1 := \left\{ \begin{pmatrix} \xi & y^\top \\ y & \tilde{X}_{1^c,1^c} \end{pmatrix} \middle| \begin{pmatrix} \xi \\ y \end{pmatrix} \in \mathbb{R}^n, \begin{pmatrix} L_{1,1} \\ L_{1^c,1} \end{pmatrix} \leq \begin{pmatrix} \xi \\ y \end{pmatrix} \leq \begin{pmatrix} U_{1,1} \\ U_{1^c,1} \end{pmatrix} \right\},$$

which implies that, for any  $p \in \{1, \dots, n\}$ ,

$$\psi_\sigma(\tilde{X}) \leq \psi_\sigma(X), \quad \forall X \in V^1 \text{ and } X_{p^c,1} = \tilde{X}_{p^c,1},$$

i.e., all components of the first row and column  $[\xi; y]$  other than the  $p$ -th are fixed. Since  $\tilde{X}$  lies in the open convex set  $S_{++}^n$ , we obtain from the optimality conditions that, for any  $p \in \{1, \dots, n\}$ ,

$$\langle \nabla \psi_\sigma(\tilde{X}), X - \tilde{X} \rangle \geq 0, \quad \forall X \in V^1 \text{ and } X_{p^c,1} = \tilde{X}_{p^c,1},$$

which further gives that, for any  $p \in \{1, \dots, n\}$ ,

$$(\nabla \psi_\sigma(\tilde{X}))_{p,1} (X_{p,1} - \tilde{X}_{p,1}) \geq 0, \quad \forall X_{p,1}$$

$$\text{such that } L_{p,1} \leq X_{p,1} \leq U_{p,1}. \quad (19.38)$$

Repeating the above argument shows that for  $i = 2, \dots, n$ , the points  $X^{k_i,i}$  also converges to  $\tilde{X}$  and

$$(\nabla \psi_\sigma(\tilde{X}))_{p,i} (X_{p,i} - \tilde{X}_{p,i}) \geq 0, \quad \forall L_{p,i} \leq X_{p,i} \leq U_{p,i}, \quad (19.39)$$

for any  $p \in \{1, \dots, n\}$ . Therefore, for any  $X \in \mathcal{X}$ , it follows from (19.38) and (19.39) that

$$\langle \nabla \psi_\sigma(\tilde{X}), X - \tilde{X} \rangle = \sum_{i,j=1,\dots,n} (\nabla \psi_\sigma(\tilde{X}))_{i,j} (X_{i,j} - \tilde{X}_{i,j}) \geq 0,$$

which implies that  $\tilde{X}$  is a global minimizer.  $\square$

## 19.4 A RBR Method for SDP with General Linear Constraints

We now consider SDP problem (19.3) with general linear constraints. Unfortunately, in this case, the RBR method may not converge to an optimal solution. This is similar to the fact that the BCD method may not converge to an optimal solution for a linearly constrained convex problem [34]. It has long been known [51] that the coordinate descent method for general nonlinear programming may not converge. Here is a 2-dimensional example that shows that for general linear constraints the RBR method may not converge to a global minimizer. Consider the following problem

$$\begin{aligned} \min \quad & X_{11} + X_{22} - \log \det(X) \\ \text{s.t.} \quad & X_{11} + X_{22} \geq 4, \quad X \geq 0. \end{aligned} \quad (19.40)$$

Starting from a point  $X$ , where  $X_{11} = 1$ ,  $X_{12} = 0$  and  $X_{22} = 3$ , the RBR subproblems are

$$\min \quad X_{11} - \log(3X_{11} - X_{12}^2), \text{ s.t. } X_{11} \geq 1,$$

and

$$\min \quad X_{22} - \log(X_{22} - X_{12}^2), \text{ s.t. } X_{22} \geq 3,$$

since  $\det(X) = X_{11}X_{22} - X_{12}^2$ . It is readily verified that optimal solutions to these subproblems are, respectively,  $X_{11} = 1$ ,  $X_{12} = 0$  and  $X_{12} = 0$ ,  $X_{22} = 3$ ; hence, the RBR method remains at the initial point, while the true optimal solution is  $X = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$ .

To overcome this type of failure, the coordinate descent method is usually applied to a sequence of unconstrained problems obtained by penalizing the constraints in the objective function. We adopt a similar approach here by embedding the pure RBR method in an augmented Lagrangian function framework. We then introduce specialized versions of this algorithm for the SDP relaxation of the maxcut problem (19.7) and the minimum nuclear norm matrix completion problem.

### 19.4.1 A RBR Augmented Lagrangian Method

In this subsection, we first introduce an augmented Lagrangian method and then combine it with the RBR method for solving the standard form SDP (19.3).

The augmented Lagrangian function for problem (19.3) taking into consideration only the general linear constraints  $\mathcal{A}(X) = b$  is defined as:

$$\mathcal{L}(X, \pi, \mu) := \langle C, X \rangle - \pi^\top (\mathcal{A}(X) - b) + \frac{1}{2\mu} \|\mathcal{A}(X) - b\|_2^2, \quad (19.41)$$

where  $\pi \in \mathbb{R}^m$  and  $\mu > 0$ . Starting from  $\pi^1 = \mathbf{0}$ ,  $\mu^1 \in (0, +\infty)$  and  $0 < \eta < 1$ , our augmented Lagrangian method iteratively solves

$$X^k := \arg \min_X \mathcal{L}(X, \pi^k, \mu^k), \quad \text{s.t. } X \succeq 0, \quad (19.42)$$

chooses  $\mu^{k+1} \in [\eta\mu^k, \mu^k]$  and then updates the vector of Lagrange multipliers by

$$\pi^{k+1} := \pi^k - \frac{\mathcal{A}(X^k) - b}{\mu^k}, \quad (19.43)$$

for the next iteration  $k + 1$ . It is important to note that our algorithm does not incorporate the positive semidefinite constraint into the augmented Lagrangian function, and therefore, it is different from the methods in [47, 78].

As is well known (see Chap. 12.2 in [25]), (19.42) is equivalent to minimizing a quadratic penalty function:

$$\begin{aligned} X^k := \arg \min_X \mathcal{F}(X, b^k, \mu^k) &:= \langle C, X \rangle \\ &\quad + \frac{1}{2\mu^k} \|\mathcal{A}(X) - b^k\|_2^2, \quad \text{s.t. } X \succeq 0, \end{aligned} \quad (19.44)$$

where  $b^k = b + \mu^k \pi^k$  and the difference between  $\mathcal{L}(X, \pi^k, \mu^k)$  and  $\mathcal{F}(X, b^k, \mu^k)$  is the constant  $-\frac{\mu^k}{2} \|\pi^k\|_2^2$ . Hence, we consider an alternative version of the augmented Lagrangian method which solves (19.44) and updates  $b^k$  by

$$b^{k+1} := b + \frac{\mu^{k+1}}{\mu^k} (b^k - \mathcal{A}(X^k)), \quad (19.45)$$

where  $b^1 := b$ . We now apply the RBR method to minimize (19.44). Starting from the point  $X^k > 0$  at the  $k$ -th iteration, the RBR subproblem corresponding to the quadratic SDP (19.44) that is obtained by fixing all elements of  $X^k$  other than those in the  $i$ -th row and column results in a minimization problem with two cone constraints. Specifically, we fix the  $n(n-1)/2$  variables in the  $(n-1) \times (n-1)$  submatrix  $B := X_{i^c, i^c}^k$  of  $X^k$  and let  $\xi$  and  $y$  denote the remaining unknown variables  $X_{i, i}$  and  $X_{i^c, i}$  (i.e., row  $i$ /column  $i$ ), respectively. Hence, the quadratic SDP problem (19.44) becomes, after, replacing the zero on the right hand side of the Schur complement constraint by  $\nu > 0$  to ensure positive definiteness of  $X^k$ ,

$$\begin{aligned} \min_{(\xi, y) \in \mathbb{R}^n} \quad & \tilde{c}^\top \begin{pmatrix} \xi \\ y \end{pmatrix} + \frac{1}{2\mu^k} \left\| \tilde{A} \begin{pmatrix} \xi \\ y \end{pmatrix} - \tilde{b} \right\|_2^2 \\ \text{s.t.} \quad & \xi - y^\top B^{-1} y \geq \nu, \end{aligned} \quad (19.46)$$

**Algorithm 6** Row-by-row augmented Lagrangian method

---

Set  $X^1 > 0$ ,  $b^1 = b$ ,  $\eta \in (0, 1)$ ,  $\nu > 0$ ,  $\mu^1 > 0$ ,  $\epsilon, \epsilon_r, \epsilon_f \geq 0$  and  $k := 1$ .  
Set  $F^0 := +\infty$  and compute  $F^1 := \langle C, X^1 \rangle$ .

**while**  $\frac{F^{k-1} - F^k}{\max\{|F^{k-1}|, 1\}} \geq \epsilon$  or  $\|\mathcal{A}(X^k) - b\|_2 \geq \epsilon_r$  **do**

Compute  $f^1 := \langle C, X^k \rangle + \frac{1}{2\mu^k} \|\mathcal{A}(X^k) - b^k\|_2^2$  and set  $f^0 := +\infty$ .

**while**  $\frac{f^{k-1} - f^k}{\max\{|f^{k-1}|, 1\}} \geq \epsilon_f$  **do**

**for**  $i = 1, \dots, n$  **do**

S1 Set  $B := X_{i^c, i^c}^k$  and compute  $\tilde{c}$ ,  $\tilde{A}$  and  $\tilde{b}$  from (19.6) with  $b = b^k$ .

S2 Solve the SOCP (19.46) and denote its solution by  $\xi$  and  $y$ .

S3 Set  $X_{i,i}^k := \xi$ ,  $X_{i^c, i}^k := y$  and  $X_{i, i^c}^k := y^\top$ .

Compute  $F^k := \langle C, X^k \rangle$  and  $f^k := F^k + \frac{1}{2\mu^k} \|\mathcal{A}(X^k) - b^k\|_2^2$ .

S4 Update  $b^{k+1} := b + \frac{\mu^{k+1}}{\mu^k} (b^k - \mathcal{A}(X^k))$ .

Choose  $\mu^{k+1} \in [\eta\mu^k, \mu^k]$  and set  $X^{k+1} := X^k$  and  $k := k + 1$ .

---

where  $\tilde{c}$ ,  $\tilde{A}$  and  $\tilde{b}$  are given by (19.6) with  $b_i$  for  $i = 1, \dots, m$  replaced by  $b_i^k$ . If we let  $LL^\top = B$  be the Cholesky factorization of  $B$  and introduce a new variable  $z = L^{-1}y$ , problem (19.46) can be written as:

$$\begin{aligned} \min_{(\xi; z; \tau)} \quad & \tilde{c}^\top \begin{pmatrix} \xi \\ Lz \end{pmatrix} + \frac{1}{2\mu} \tau \\ \text{s.t.} \quad & \left\| \begin{pmatrix} \tilde{A} & \xi \\ Lz & \end{pmatrix} - \tilde{b} \right\|_2^2 \leq \tau \\ & \|z\|_2^2 \leq \xi - \nu. \end{aligned} \tag{19.47}$$

Therefore, each step of our RBR augmented Lagrangian method involves solving a SOCP with two rotated second-order cone constraints. We plan to show how advantage can be taken of the particular form of these SOCPs in a future paper. If  $B$  is only positive semidefinite, we can derive a similar SOCP by using the spectral decomposition of  $B$ . For references on solving SOCPs, see [1] for example. Our combined RBR augmented Lagrangian method for minimizing (19.3) is presented in Algorithm 6.

The RBR method applied to problem (19.44) converges by Theorem 19.2 since solving the RBR subproblem (19.46) essentially corresponds to minimizing the unconstrained function obtained by subtracting  $\sigma \log(\xi - y^\top B^{-1}y)$  from the objective function in (19.46) using an argument analogous to the one made in Sect. 19.3.4. It is well known that an augmented Lagrangian method applied to minimizing a strictly convex function subject to linear equality constraints, where the minimization of

the augmented Lagrangian for each value of the multiplier  $\lambda^k$  ( $b^k$  in Algorithm 6) is either done exactly or is asymptotically exact, converges to an optimal solution [4,5,52]. Hence, it is clear that a slightly modified version of Algorithm 2 converges to such a solution. For more details for the exact minimization case, we refer the reader to [66].

### 19.4.2 Application to Maxcut SDP

Since the constraints in problem (19.7) are  $X_{i,i} = 1$  for  $i = 1, \dots, n$ , the quadratic term in the objective function of the RBR subproblem simplifies to

$$\left\| \tilde{A} \begin{pmatrix} \xi \\ y \end{pmatrix} - \tilde{b} \right\|^2 = (\xi - b_i^k)^2,$$

and problem (19.46) reduces to

$$\begin{aligned} \min_{(\xi, y) \in \mathbb{R}^n} \quad & c\xi + \tilde{c}^\top y + \frac{1}{2\mu^k} (\xi - b_i^k)^2 \\ \text{s.t.} \quad & \xi - y^\top B^{-1} y \geq \nu, \end{aligned} \quad (19.48)$$

where  $c := C_{i,i}$ ,  $\tilde{c} := 2C_{i^*, i}$  and  $b_i^1 = 1$ . The first-order optimality conditions for (19.48) are

$$\begin{aligned} \xi &= b_i^k + \mu^k(\lambda - c), \quad y = -\frac{1}{2\lambda} B \tilde{c} \\ \xi &\geq y^\top B^{-1} y + \nu, \quad \lambda \geq 0 \quad \text{and} \quad (\xi - y^\top B^{-1} y - \nu)\lambda = 0. \end{aligned}$$

If  $\tilde{c} = 0$ , then  $y = 0$  and  $\xi = \max\{v, b_i^k - \mu^k c\}$ . Otherwise,  $\lambda$  is the unique real root of the cubic equation:

$$\varphi(\lambda) := 4\mu^k \lambda^3 + 4(b_i^k - \mu^k c - \nu)\lambda^2 - \gamma = 0, \quad (19.49)$$

which is positive. This follows from the continuity of  $\varphi(\lambda)$  and the facts that  $\varphi(0) = -\tilde{c}^\top B \tilde{c} < 0$ ,  $\lim_{\lambda \rightarrow +\infty} \varphi(\lambda) = +\infty$  and

$$\varphi'(\lambda) = 12\mu^k \lambda^2 + 8(b_i^k - \mu^k c - \nu)\lambda \geq 4\mu^k \lambda^2$$

since  $\xi = b_i^k - \mu^k c + \mu^k \lambda \geq \nu$ , which implies that  $\varphi'(0) = 0$  and  $\varphi'(\lambda) > 0$  for  $\lambda \neq 0$ . The RBR augmented Lagrangian method for problem (19.7) is denoted by ALAG-RBR-M.

### 19.4.3 Application to Matrix Completion SDP

Using the notation from Sect. 19.3.2, the norm of the constraint residual of each RBR subproblem (19.46) is  $\left\| \tilde{A} \begin{pmatrix} \xi \\ y \end{pmatrix} - \tilde{b} \right\| = \left\| X_{\alpha,i} - \tilde{b} \right\| =: \|\hat{y} - \tilde{b}\|$ , where

$$\tilde{b} := \begin{cases} (M_{i,\bar{\alpha}}^k)^T, & \text{if } i \leq p, \\ M_{\alpha,i-p}^k, & \text{if } p < i \leq n, \end{cases} \quad (19.50)$$

and  $M^1 = M$ . Therefore, the SOCP (19.46) becomes

$$\begin{aligned} \min_{(\xi;y) \in \mathbb{R}^n} \quad & \xi + \frac{1}{2\mu^k} \|\hat{y} - \tilde{b}\|_2^2 \\ \text{s.t.} \quad & \xi - y^\top B^{-1} y \geq v, \end{aligned} \quad (19.51)$$

where the matrix  $B = \begin{pmatrix} X_{\alpha,\alpha}^k & X_{\alpha,\beta}^k \\ X_{\beta,\alpha}^k & X_{\beta,\beta}^k \end{pmatrix}$ .

The optimal solution  $[\xi; y] = [\xi; \hat{y}; \tilde{y}]$  of (19.51) must satisfy  $\xi = y^\top B^{-1} y + v$ . Hence, (19.51) is equivalent to an unconstrained quadratic minimization problem

$$\min_y y^\top B^{-1} y + \frac{1}{2\mu^k} \|\hat{y} - \tilde{b}\|_2^2, \quad (19.52)$$

whose optimality conditions are

$$\begin{pmatrix} X_{\alpha,\alpha}^k & X_{\alpha,\beta}^k \\ X_{\beta,\alpha}^k & X_{\beta,\beta}^k \end{pmatrix}^{-1} \begin{pmatrix} \hat{y} \\ \tilde{y} \end{pmatrix} + \frac{1}{2\mu^k} \begin{pmatrix} \hat{y} - \tilde{b} \\ \mathbf{0} \end{pmatrix} = 0. \quad (19.53)$$

which implies that

$$\begin{pmatrix} \hat{y} \\ \tilde{y} \end{pmatrix} + \frac{1}{2\mu^k} \begin{pmatrix} X_{\alpha,\alpha}^k \\ X_{\beta,\alpha}^k \end{pmatrix} \hat{y} = \frac{1}{2\mu^k} \begin{pmatrix} X_{\alpha,\alpha}^k \\ X_{\beta,\alpha}^k \end{pmatrix} \tilde{b}.$$

Solving for  $\hat{y}$  and  $\tilde{y}$  we obtain  $\tilde{y} = \frac{1}{2\mu^k} X_{\beta,\alpha}^k (\tilde{b} - \hat{y})$ , where  $\hat{y}$  can be computed from the system of linear equations  $(2\mu^k I + X_{\alpha,\alpha}^k) \hat{y} = X_{\alpha,\alpha}^k \tilde{b}$ . Then, it follows from  $\xi = y^\top B^{-1} y + v$  and (19.53) that  $\xi = \frac{1}{2\mu^k} \hat{y}^\top (\tilde{b} - \hat{y}) + v$ .

The above specialized augmented Lagrangian RBR method for minimizing (19.20) is denoted by RBR-MC. As in the pure RBR method for the matrix completion problem, we only need to solve a single system of linear equations, whose size is expected to be small for each RBR subproblem (19.51).

We note that the subproblems that arise when the augmented Lagrangian version of the RBR method is applied to several other SDP problems are also solvable in closed form as in the computation of the Lovasz theta function. We did not include a discussion of this or of other SDPs, such as the theta plus problem, that result in subproblems that are rather special quadratic programs, and hence efficiently solvable, to keep the length of this chapter reasonable.

## 19.5 An Extension of RBR Using Rank-Two Updates

In this section, we propose a generalization of the RBR scheme that uses rank-two updates besides those that correspond to modifying a single row and column. This results in a method that also requires solving a sequence of SOCPs. Specifically, given a positive definite matrix  $X > 0$  and vectors  $u, v \in \mathbb{R}^n$ , we consider the rank-two update:

$$X_+ = X + \frac{1}{2}(uv^\top + vu^\top). \quad (19.54)$$

The RBR scheme is a special case of (19.54), corresponding to  $u = e_i$ , where  $e_i$  is the  $i$ -th column of the identity matrix.

By allowing the algorithm to consider different vectors  $u$  we significantly increase the flexibility of the BCD scheme to exploit the problem structure. For instance, if  $u = (e_i + e_j)/2$  then the BCD will modify the  $(i, j)$ -th pair of rows and columns simultaneously. This can be useful, for instance, when the linear constraints are of the form  $X_{ii} + X_{jj} - 2X_{ij} = d_{ij}$  as occurs in SDP relaxations of sensor network localization problems. More generally, one might want to update a whole block of columns and rows at a time because the variables defined in the blocks are closely related to each other via some constraints. For instance, in sparse inverse covariance selection some of the random variables may be known to be directly correlated, hence it makes sense to update the corresponding rows and columns of the covariance matrix in a related manner. In the case of the sensor network localization problem the network may consist of several loosely connected small clusters, for each of which the distance structure is highly constrained. In this case it also makes sense to update the rows and columns related to the whole cluster rather than to individual sensors, while preserving the constraints for the cluster by choosing an appropriate  $u$  which keeps the step  $uv^\top + vu^\top$  in the nullspace of the chosen subset of constraints.

Alternatively, one may choose  $u$  to be the leading eigenvector of the objective function gradient, hence including the steepest descent rank-two direction into the range of possible BCD steps. While a numerically efficient choice of  $u$  is likely to be tailored to the specific SDP problem being solved, here we consider the general case.

The positive definiteness of  $X_+$  in (19.54) can be expressed as a second-order cone constraint for any fixed vector  $u$ , given that  $X$  is positive definite. To see this, let  $X = LL^\top$  be the Cholesky factorization of  $X$ , where  $L$  is a lower triangular matrix. If

we define  $y = L^{-1}u$  and  $x = L^{-1}v$ , then the matrix  $X_+$  can be factorized as  $X_+ = LVL^\top$ , where  $V := I + \frac{1}{2}(yx^\top + xy^\top)$ . It can be easily verified that  $z_1 := \|y\|_2 x - \|x\|_2 y$  and  $z_2 := \|y\|_2 x + \|x\|_2 y$  are eigenvectors of  $V$  corresponding to the eigenvalues

$$\lambda_1 := 1 + \frac{1}{2}y^\top x - \frac{1}{2}\|y\|_2\|x\|_2 \quad \text{and} \quad \lambda_2 := 1 + \frac{1}{2}y^\top x + \frac{1}{2}\|y\|_2\|x\|_2, \quad (19.55)$$

respectively. The eigenvalues other than  $\lambda_1$  and  $\lambda_2$  are equal to 1 since  $V$  is a rank-two update of the identity. Hence, the matrix  $V$  is positive definite if

$$\lambda_1 = 1 + \frac{1}{2}y^\top x - \frac{1}{2}\|y\|_2\|x\|_2 > 0, \quad (19.56)$$

which is equivalent to  $2 + u^\top X^{-1}v - \sqrt{(u^\top X^{-1}u)(v^\top X^{-1}v)} > 0$ . Since  $X_+$  in (19.54) can be written as  $X_+ = X + \bar{u}\bar{u}^\top - \bar{v}\bar{v}^\top$ , where  $\bar{u} = \frac{1}{2}(u+v)$  and  $\bar{v} = \frac{1}{2}(u-v)$ , the Cholesky factorization of  $X_+$  can be obtained in  $O(n^2)$  operations from two rank-1 updates to the Cholesky factorization of  $X$ .

As in the augmented Lagrangian RBR approach for solving (19.3) described in Sect. 19.4.1, we can incorporate the rank-two update in an augmented Lagrangian framework. Our goal is to solve (19.44) by iteratively solving subproblems generated by our rank-two updating technique. Given a matrix  $X > 0$  and a vector  $u$ , substituting  $X_+$  for  $X$  in (19.44) and using (19.54) and (19.56), we obtain the subproblem

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \varphi(x) := c^\top x + \frac{1}{2\mu^k} \|Bx - d\|_2^2 \\ \text{s.t.} \quad & 2 + y^\top x - \|y\|_2\|x\|_2 \geq \sigma, \end{aligned} \quad (19.57)$$

where

$$y = L^{-1}u, \quad c := L^\top Cu, \quad B := \begin{pmatrix} u^\top A^{(1)}L \\ \dots \\ u^\top A^{(m)}L \end{pmatrix}, \quad d = b^k - \mathcal{A}(X), \quad (19.58)$$

for finding  $v = Lx$ . Note that the subproblem (19.57) can be formulated as an SOCP with two second-order cone constraints.

In general the matrix  $B$  has  $m$  rows. The  $l$ -th row of  $B$  is  $u^\top A^{(l)}L$  and hence is equal to 0 if  $u^\top A^{(l)} = 0$ . As discussed above,  $u$  can be chosen to contain only a few nonzeros. For instance, when  $u = e_i + e_j$ , the only rows of  $B$  that are nonzero are those corresponding to  $A^{(l)}$  that have nonzero elements in row  $i$  or  $j$ . In particular, in sensor network localization problems the number of rows in  $B$  will equal the number of links that involve nodes  $i$  or  $j$ ; hence, the size of the SOCP cone in the objective function in the subproblem (19.57) will often be much smaller than the total number of constraints.

It seems straightforward to extend the convergence result stated in Theorem 19.2 for optimizing the log det analog of Problem (19.57) to the case of rank-two updates. The theory easily extends if the set of rank-two updates is defined by a finite set of directions  $u_i$ , which span  $R^n$  and through which the algorithm cycles (as in the case of RBR, where  $u_i = e_i$ ). More generally we can allow an infinite set of directions, but only under some additional restrictions. For instance one such restriction is that the set of limit points of the set of directions is finite and spans  $R^n$ . A suitable choice for the set of possible directions is likely to depend on the particular application and is subject to further study.

## 19.6 Numerical Results

Although the numerical results that we present in this section are limited to two special classes of SDP problems, they illustrate the effectiveness of our RBR algorithmic framework when it gives rise to easily solved subproblems and our tests show that the number of cycles taken by our algorithms grows very slowly with the size of the problem. Hence, in these cases, large scale SDPs can be solved in a moderate amount of time using only moderate amount of memory. In fact, approximate solutions even with a relatively low accuracy can be quite helpful for the solution of SDP relaxations of combinatorial problems, especially if a sequence of relaxations has to be solved and updated in branch-and-bound or column-generation schemes. In such situations, the computation of high accuracy solutions may be unnecessary, especially in early stages in which the bounds are still relatively weak and these solutions are only used for branching decisions or the generation of cutting planes. Furthermore, because optimal objective values of combinatorial problems are typically integer valued, so that upper (lower) SDP bounds can be rounded down (up), the importance of a highly accurate solution is again rather questionable.

### 19.6.1 The Maxcut SDP Relaxation

In this subsection, we demonstrate the effectiveness of the RBR methods PURE-RBR-M and ALAG-RBR-M on a set of maxcut SDP relaxation problems and compare them with the code “maxcut” in DSDP (version 5.8) [3] and a routine in SDPLR (version 0.130301) [13] developed especially for the maxcut SDP. The DSDP code implements a dual interior point method that is designed to take advantage of the structure of such problems. The SDPLR code implements a low-rank factorization approach. The main parts of our code were written in C Language MEX-files in MATLAB (Release 7.3.0), and all experiments were performed on a Dell Precision 670 workstation with an Intel Xeon 3.4GHZ CPU and 6GB of RAM.

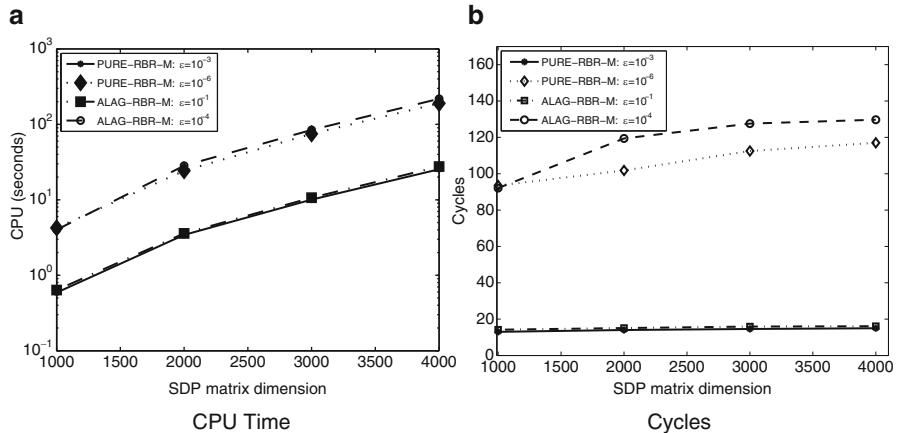
**Table 19.1** Computational results for the maxcut SDP relaxation

Name	DSDP obj	CPU	SDPLR rel-obj	CPU	PURE-RBR-M			PURE-RBR-M		
					$\epsilon = 10^{-3}$ rel-obj	CPU	cycle	$\epsilon = 10^{-6}$ rel-obj	CPU	cycle
R1000-1	-1.4e+3	52.6	1.6e-05	4.0	4.9e-3	0.6	13	3.0e-5	3.9	90
R1000-2	-1.4e+3	57.0	6.5e-06	6.0	5.0e-3	0.6	13	3.6e-5	4.1	96
R2000-1	-4.1e+3	607.6	5.1e-05	18.8	5.0e-3	3.9	14	3.7e-5	26.5	97
R2000-2	-4.1e+3	602.3	5.8e-05	19.4	5.2e-3	3.9	14	3.6e-5	27.5	101
R3000-1	-7.7e+3	2576	4.2e-05	38.2	5.0e-3	12.8	15	4.1e-5	90.0	103
R3000-2	-7.7e+3	2606	4.3e-05	42.2	5.2e-3	13.2	15	3.7e-5	89.4	105
R4000-1	-1.2e+4	6274	6.9e-05	64.3	5.9e-3	36.5	15	4.0e-5	261.1	108
R4000-2	-1.2e+4	6310	6.6e-05	63.4	5.7e-3	36.3	15	3.9e-5	265.7	108
Random planar graphs										
P1000-1	-1.4e+3	45.1	1.5e-05	6.3	5.0e-3	0.6	13	4.0e-5	4.9	102
P1000-2	-1.4e+3	45.5	8.9e-06	4.6	4.4e-3	0.6	13	2.9e-5	4.2	89
P2000-1	-2.9e+3	386.1	4.4e-06	43.9	5.5e-3	3.0	14	3.7e-5	21.6	102
P2000-2	-2.8e+3	362.8	5.7e-05	19.2	5.8e-3	2.9	14	3.9e-5	22.1	109
P3000-1	-4.3e+3	1400	1.1e-05	49.9	6.0e-3	7.3	15	4.0e-5	56.3	117
P3000-2	-4.3e+3	1394	1.4e-05	62.3	6.5e-3	7.0	14	4.7e-5	57.2	119
P4000-1	-5.7e+3	3688	1.5e-05	122.7	6.5e-3	14.3	15	4.3e-5	114.2	124
P4000-2	-5.9e+3	3253	9.9e-06	123.9	6.5e-3	14.4	15	4.9e-5	116.7	126

The test problems are based on graphs generated by “rudy”, a machine independent graph generator written by G.Rinaldi. Details of the generation including the arguments of “rudy” are provided in [67]. The tolerance in the code DSDP was set to  $1e-3$ . The tolerance in the code SDPLR was set to  $2e-5$  and the parameter file “p.maxcut5” was used. The parameter  $v$  in the RBR methods was set to  $10^{-6}$ . We ran PURE-RBR-M with two different tolerances, i.e.,  $\epsilon$  was set to  $10^{-3}$  (moderately accurate) and  $10^{-6}$  (highly accurate), respectively. Similarly, we ran ALAG-RBR-M with two different tolerance settings, that is,  $\epsilon, \epsilon_r, \epsilon_f$  were all set to  $10^{-1}$  and  $10^{-4}$ , respectively. For practical considerations, we terminated minimizing each augmented Lagrangian function if the number of cycles was greater than 5. The initial penalty parameter  $\mu^1$  in ALAG-RBR-M was set to 5 and was updated by  $\mu^{k+1} = \max(0.5\mu^k, 10^{-1})$ .

A summary of the computational results obtained by DSDP, SDPLR and PURE-RBR-M is presented in Table 19.1. In the table, “obj” denotes the objective function value of the dual problem computed by DSDP, “rel-obj” denotes the relative error between “obj” and the objective function value computed by either the RBR methods or SDPLR, “CPU” denotes CPU time measured in seconds, and “cycle” denotes the total number of RBR cycles. From Table 19.1, we can see that our RBR code is able to solve the maxcut SDP relaxation very efficiently. The number of cycles required was almost the same for all of the problems, no matter what their size was. The RBR method was also quite competitive with SDPLR in achieving a relative accuracy of roughly  $5 \times 10^{-5}$  in the objective function value.

To illustrate the relationship between the computational cost of the RBR methods and the dimension of the SDP matrices, we plot the average of the CPU time versus the dimension in Fig. 19.1a and the average of the number of cycles versus the dimension in Fig. 19.1b. Somewhat surprisingly, these plots show that the



**Fig. 19.1** Relationship between the computational cost and SDP matrix dimension for the maxcut SDP relaxation. (a) CPU time. (b) Cycles

augmented Lagrangian RBR algorithm solved the maxcut SDP problems almost as efficiently as the pure RBR algorithm for a given relative error. Consequently we did not include test results for ALAG-RBR-M in Table 19.1.

### 19.6.2 Matrix Completion

In this subsection, we evaluate the augmented Lagrangian version of the RBR method (RBR-MC) for the matrix completion problem (19.20). While the pure RBR method can be directly applied to this problem, preliminary numerical testing showed that this approach is much slower (i.e., converges much more slowly) than using RBR-MC, which requires only a small amount of additional work to solve each subproblem than the pure method.

It seems that the pure RBR method gets trapped close to the boundary of the semidefinite cone. To overcome this we also tried starting with a very large value of  $v$  (say  $v = 100$ ), reducing  $v$  every 20 cycles by a factor of 4 until it reached a value of  $10^{-6}$ . While this improved the performance of the method, the augmented Lagrangian version was still two to four times faster. Hence, we only present results for the latter method. Although we compare RBR-MC with the specialized algorithms, such as SVT [40] and FPCA [46], for the matrix completion problem (19.19), our main purpose here is to demonstrate that the RBR method can efficiently solve the SDP problem (19.20) rather than to compete with those latter algorithms. In fact, the solver LMaFit [69] was consistently much faster than all the methods mentioned above. DSDP is not included in this comparison because it takes too long to solve all problems.

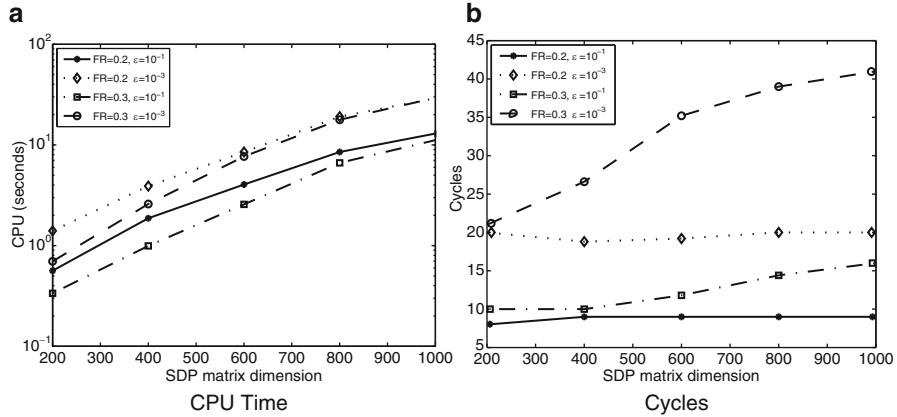
**Table 19.2** Computational results for the matrix completion problem with FR=0.2

seed	RBR-MC ( $\epsilon = 10^{-1}$ )			RBR-MC ( $\epsilon = 10^{-3}$ )			FPCA		SVT	
	rel-X	CPU	cycle	rel-X	CPU	cycle	rel-X	CPU	rel-X	CPU
p=q=200; r=10; m=19500; SR=0.49										
68521	7.5e-05	1.8	9	9.4e-07	3.6	17	1.6e-06	2.7	1.6e-05	20.4
56479	6.0e-05	1.9	9	7.4e-07	3.4	17	1.5e-06	2.7	1.6e-05	13.6
p=q=300; r=10; m=29500; SR=0.33										
68521	1.0e-04	3.9	9	1.5e-06	7.2	17	2.2e-06	4.6	1.7e-05	28.2
56479	1.0e-04	4.0	9	1.7e-06	7.5	17	2.1e-06	4.6	1.7e-05	39.0
p=q=400; r=10; m=39500; SR=0.25										
68521	1.0e-04	7.6	9	2.1e-06	14.3	17	2.8e-06	6.0	1.8e-05	28.8
56479	9.9e-05	5.7	9	1.9e-06	10.9	17	2.9e-06	6.0	1.7e-05	28.1
p=q=500; r=10; m=49500; SR=0.20										
68521	2.4e-04	8.8	9	1.8e-06	18.6	19	3.6e-06	10.0	1.9e-05	49.0
56479	1.1e-04	9.0	9	1.5e-06	19.0	19	3.8e-06	10.0	1.8e-05	45.9

**Table 19.3** Computational results for the matrix completion problem with FR=0.3

seed	RBR-MC ( $\epsilon = 10^{-1}$ )			RBR-MC ( $\epsilon = 10^{-3}$ )			FPCA		SVT	
	rel-X	CPU	cycle	rel-X	CPU	cycle	rel-X	CPU	rel-X	CPU
p=q=200; r=10; m=13000; SR=0.33										
68521	1.0e-03	1.0	10	4.4e-06	2.3	24	3.3e-06	8.4	5.9e-04	96.7
56479	1.5e-03	1.0	10	6.8e-06	2.3	24	3.0e-06	8.5	1.3e-03	88.7
p=q=300; r=10; m=19666; SR=0.22										
68521	1.0e-03	2.4	11	3.7e-06	5.5	26	3.4e-06	4.9	2.7e-03	180.6
56479	3.3e-04	2.6	12	2.2e-06	5.9	27	3.6e-06	4.5	2.5e-03	230.8
p=q=400; r=10; m=26333; SR=0.16										
68521	9.8e-03	6.1	15	9.9e-04	16.2	40	4.8e-06	10.4	2.8e-02	418.8
56479	3.0e-03	5.6	14	4.6e-06	11.2	28	4.0e-06	16.1	1.5e-02	374.7
p=q=500; r=10; m=33000; SR=0.13										
68521	5.3e-03	11.4	16	6.9e-06	20.1	29	6.1e-06	223.4	2.7e-02	675.0
56479	5.4e-03	11.0	16	5.9e-06	21.3	32	6.2e-06	212.8	2.8e-02	667.4

Random matrices  $M \in \mathbb{R}^{p \times q}$  with rank  $r$  were created by the procedure in [46]. The ratio  $m/(pq)$  between the number of measurements and the number of entries in  $M$  is denoted by “SR” (sampling ratio). The ratio  $r(p+q-r)/m$  of the dimension of a rank  $r$  matrix to the number of samples is denoted by “FR”. In our tests, the rank  $r$  and the number of sampling entries  $m$  were taken consistently so that according to the theory in [16] the matrix  $M$  is the optimal solution of problem (19.20). Specifically, FR was set to 0.2 and 0.3 and  $r$  was set to 10. We tested five square matrices  $M$  with dimensions  $p = q \in \{200, \dots, 500\}$  and set the number  $m$  to  $r(p+q-r)/\text{FR}$ . All parameters  $p, q, r, m$  and the random seeds “seed” used by the random number generators “rand” and “randn” in MATLAB are reported in Tables 19.2 and 19.3.



**Fig. 19.2** Relationship between the computational cost and SDP matrix dimension for SDP matrix completion. **(a)** CPU time. **(b)** Cycles

We ran RBR-MC with two different tolerance settings, i.e.,  $\epsilon, \epsilon_r, \epsilon_f$  were all set to  $10^{-1}$  and  $10^{-3}$ , respectively. All other parameters of RBR-MC were set to the same values as those used in ALAG-RBR-M. The tolerance parameter “xtol” of FPCA was set to  $10^{-6}$  and all other parameters were set to their default values. We tried many different parameter settings but could not get SVT to work well on all problems. Hence, we only report the results of SVT for the “best” parameter setting that we found, i.e., the parameters “tau” and “delta” and “tol” were set to  $5n$ ,  $\min(\max(1.2n^2/p, 1), 3)$  and  $10^{-5}$ , respectively. Summaries of the computational results for FR=0.2 and FR = 0.3 are presented in Tables 19.2 and 19.3, respectively. In these tables,  $\text{rel-}X := \frac{\|X - M\|_F}{\|M\|_F}$  gives the relative error between the true and the recovered matrices. From these tables we see that to compute a solution of comparable accuracy, RBR-MC (with  $\epsilon = 10^{-3}$ ) was faster than FPCA when the SDP matrix dimension was small. When that dimension was larger FPCA was typically as fast as RBR-MC, but not more than approximately twice as fast. However, there is an exception to this in that FPCA took from 20 to 35 times as much CPU time to solve the examples with  $p = q = 500$  when FR=0.3 as did the RBR method. To illustrate the relationship between the computational cost of the RBR method and the dimension of the matrices, we plot the average of the CPU time versus the dimension of the SDP matrix (i.e.,  $p + q$ ) in Fig. 19.2a and the average of the number of cycles versus this dimension in Fig. 19.2b.

## 19.7 Summary

In this chapter, we have shown that RBR block coordinate descent methods can be very effective for solving certain SDPs. In particular, they work extremely well when the subproblem that needs to be solved for each block of variables can be given

in closed form, as in SDPs that arise as relaxations of maxcut, matrix completion and Lovasz theta function problems. The RBR method is also effective when the RBR subproblems can be formulated as simple quadratic programming problems, as in sparse inverse covariance estimation, the computation of the Lovasz theta plus function, and relaxations of the maximum k-cut and bisection problems.

Like all first-order methods, they are best suited to situations where highly accurate solutions are not required. As is the case for BCD and coordinate descent methods in general, only constraints that do not couple different variable blocks can be handled directly. For more general linear constraints, the RBR approach has to be incorporated into an augmented Lagrangian framework. Our numerical testing has shown that even problems in which the constraints do not couple variables from different blocks, it still may be advantageous to employ an augmented Lagrangian approach, since this gives the method more freedom of movement. In addition, starting very close to the boundary of the semidefinite cone, especially when combined with linear constraints that very tightly limit the size of steps that the RBR method can take, can result in very slow rates of convergence.

Finally, we have discussed how the RBR approach can be generalized to accommodate rank-two updates other than those that correspond to modifying a single row and column.

**Acknowledgements** The research presented in this chapter was supported in part by NSF Grants DMS-0439872, DMS 06-06712, DMS 10-16571, ONR Grant N00014-08-1-1118 and DOE Grant DE-FG02-08ER58562. The authors would like to thank Shiqian Ma for his help in writing and testing the codes, especially for the matrix completion problem, and the editors and two anonymous referees for their valuable comments and suggestions.

## References

1. Alizadeh, F., Goldfarb, D.: Second-order cone programming. *Math. Programming* **95**(1, Ser. B), 3–51 (2003). ISMP 2000, Part 3 (Atlanta, GA)
2. Banerjee, O., El Ghaoui, L., d'Aspremont, A.: Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *J. Mach. Learn. Res.* **9**, 485–516 (2008)
3. Benson, S.J., Ye, Y., Zhang, X.: Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM J. Optim.* **10**(2), 443–461 (2000)
4. Bertsekas, D.P.: Necessary and sufficient condition for a penalty method to be exact. *Math. Programming* **9**(1), 87–99 (1975)
5. Bertsekas, D.P.: Constrained optimization and Lagrange multiplier methods. Computer Science and Applied Mathematics. Academic Press Inc., New York (1982)
6. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific (1999)
7. Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and distributed computation: numerical methods. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1989)
8. Bo, L., Sminchisescu, C.: Greedy Block Coordinate Descent for Large Scale Gaussian Process Regression. In: Uncertainty in Artificial Intelligence (2008)
9. Bouman, C.A., Sauer, K.: A unified approach to statistical tomography using coordinate descent optimization. *IEEE Transactions on Image Processing* **5**, 480–492 (1996)

10. Boyd, S., Vandenberghe, L.: Convex optimization. Cambridge University Press, Cambridge (2004)
11. Burer, S.: Optimizing a polyhedral-semidefinite relaxation of completely positive programs. Tech. rep., Department of Management Sciences, University of Iowa (2008)
12. Burer, S., Monteiro, R.D.C.: A projected gradient algorithm for solving the maxcut SDP relaxation. *Optim. Methods Softw.* **15**(3-4), 175–200 (2001)
13. Burer, S., Monteiro, R.D.C.: A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Math. Program.* **95**(2, Ser. B), 329–357 (2003)
14. Burer, S., Monteiro, R.D.C.: Local minima and convergence in low-rank semidefinite programming. *Math. Program.* **103**(3, Ser. A), 427–444 (2005)
15. Burer, S., Vandenbussche, D.: Solving lift-and-project relaxations of binary integer programs. *SIAM J. Optim.* **16**(3), 726–750 (2006)
16. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. *Foundations of Computational Mathematics* (2009). DOI 10.1007/s10208-009-9045-5
17. Chang, K.W., Hsieh, C.J., Lin, C.J.: Coordinate descent method for large-scale L2-loss linear support vector machines. *J. Mach. Learn. Res.* **9**, 1369–1398 (2008)
18. Chen, G., Teboulle, M.: A proximal-based decomposition method for convex minimization problems. *Math. Programming* **64**(1, Ser. A), 81–101 (1994)
19. Clarkson, K.L.: Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. In: *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 922–931 (2008)
20. Diachin, L.F., Knupp, P., Munson, T., Shontz, S.: A comparison of inexact newton and coordinate descent mesh optimization techniques. In: *13th International Meshing Roundtable*, Williamsburg, VA, Sandia National Laboratories, pp. 243–254 (2004)
21. Eckstein, J., Bertsekas, D.P.: An alternating direction method for linear programming. LIDS-P, 1967. Laboratory for Information and Decision Systems, MIT
22. Eckstein, J., Bertsekas, D.P.: On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Programming* **55**(3, Ser. A), 293–318 (1992)
23. Elad, M., Matalon, B., Zibulevsky, M.: Coordinate and subspace optimization methods for linear least squares with non-quadratic regularization. *Appl. Comput. Harmon. Anal.* **23**(3), 346–367 (2007)
24. Fessler, J.A.: Grouped coordinate descent algorithms for robust edge-preserving image restoration. In: *in Proc. SPIE 3071, Im. Recon. and Restor. II*, pp. 184–94 (1997)
25. Fletcher, R.: Practical methods of optimization, second edn. A Wiley-Interscience Publication. John Wiley & Sons Ltd., Chichester (1987)
26. Friedman, J., Hastie, T., Tibshirani, R.: Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9**, 432–441 (2008)
27. Fujisawa, K., Kojima, M., Nakata, K.: Exploiting sparsity in primal–dual interior-point methods for semidefinite programming. *Math. Programming* **79**(1-3, Ser. B), 235–253 (1997)
28. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.* **42**(6), 1115–1145 (1995)
29. Goldfarb, D., Iyengar, G.: Robust convex quadratically constrained programs. *Math. Program.* **97**(3, Ser. B), 495–515 (2003). New trends in optimization and computational algorithms (NTOC 2001) (Kyoto)
30. Goldfarb, D., Iyengar, G.: Robust portfolio selection problems. *Math. Oper. Res.* **28**(1), 1–38 (2003)
31. Goldfarb, D., Ma, S.: Fast alternating linearization methods for minimizing the sum of two convex functions. Tech. rep., IEOR, Columbia University (2009)
32. Goldfarb, D., Ma, S.: Fast multiple splitting algorithms for convex optimization. Tech. rep., IEOR, Columbia University (2009)
33. Grippo, L., Sciandrone, M.: Globally convergent block-coordinate techniques for unconstrained optimization. *Optim. Methods Softw.* **10**(4), 587–637 (1999)

34. Grippo, L., Sciandrone, M.: On the convergence of the block nonlinear Gauss-Seidel method under convex constraints. *Oper. Res. Lett.* **26**(3), 127–136 (2000)
35. Hackbusch, W.: Multigrid methods and applications, *Springer Series in Computational Mathematics*, vol. 4. Springer-Verlag, Berlin (1985)
36. He, B., Liao, L.Z., Han, D., Yang, H.: A new inexact alternating directions method for monotone variational inequalities. *Math. Program.* **92**(1, Ser. A), 103–118 (2002)
37. He, B.S., Yang, H., Wang, S.L.: Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *J. Optim. Theory Appl.* **106**(2), 337–356 (2000)
38. Helmberg, C., Rendl, F.: A spectral bundle method for semidefinite programming. *SIAM J. Optim.* **10**(3), 673–696 (2000)
39. Huang, F.L., Hsieh, C.J., Chang, K.W., Lin, C.J.: Iterative scaling and coordinate descent methods for maximum entropy. In: ACL-IJCNLP '09: Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pp. 285–288. Association for Computational Linguistics, Morristown, NJ, USA (2009)
40. Jian-Feng, C., Candes, E.J., Zuowei, S.: A singular value thresholding algorithm for matrix completion export. *SIAM J. Optim.* **20**, 1956–1982 (2010)
41. Kiwiel, K.C., Rosa, C.H., Ruszczyński, A.: Proximal decomposition via alternating linearization. *SIAM J. Optim.* **9**(3), 668–689 (1999)
42. Kolda, T.G., Lewis, R.M., Torczon, V.: Optimization by direct search: new perspectives on some classical and modern methods. *SIAM Rev.* **45**(3), 385–482 (2003)
43. Kontogiorgis, S., Meyer, R.R.: A variable-penalty alternating directions method for convex optimization. *Math. Programming* **83**(1, Ser. A), 29–53 (1998)
44. Li, Y., Osher, S.: Coordinate descent optimization for  $\ell^1$  minimization with application to compressed sensing: a greedy algorithm. *Inverse Probl. Imaging* **3**(3), 487–503 (2009)
45. Luo, Z.Q., Tseng, P.: On the convergence of the coordinate descent method for convex differentiable minimization. *J. Optim. Theory Appl.* **72**(1), 7–35 (1992)
46. Ma, S., Goldfarb, D., Chen, L.: Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming* pp. 1–33 (2009)
47. Malick, J., Povh, J., Rendl, F., Wiegele, A.: Regularization methods for semidefinite programming. *SIAM Journal on Optimization* **20**(1), 336–356 (2009)
48. Mazumder, R., Friedman, J., Hastie, T.: Sparsenet: Coordinate descent with non-convex penalties. Tech. rep., Stanford University (2009)
49. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. Tech. rep., CORE Discussion Paper (2010)
50. Povh, J., Rendl, F., Wiegele, A.: A boundary point method to solve semidefinite programs. *Computing* **78**(3), 277–286 (2006)
51. Powell, M.J.D.: On search directions for minimization algorithms. *Math. Programming* **4**, 193–201 (1973)
52. Rockafellar, R.T.: Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Math. Oper. Res.* **1**(2), 97–116 (1976)
53. Saad, Y.: Iterative methods for sparse linear systems, second edn. Society for Industrial and Applied Mathematics, Philadelphia, PA (2003)
54. Saha, A., Tewari, A.: On the finite time convergence of cyclic coordinate descent methods. Tech. rep., University of Chicago (2010)
55. Shalev-Shwartz, S., Tewari, A.: Stochastic methods for  $\ell_1$  regularized loss minimization. In: ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 929–936. ACM (2009)
56. Tai, X.C., Xu, J.: Global and uniform convergence of subspace correction methods for some convex optimization problems. *Math. Comp.* **71**(237), 105–124 (2002)
57. Todd, M.J.: Semidefinite optimization. *Acta Numer.* **10**, 515–560 (2001)
58. Tseng, P.: Dual coordinate ascent methods for non-strictly convex minimization. *Math. Programming* **59**(2, Ser. A), 231–247 (1993)

59. Tseng, P.: Alternating projection-proximal methods for convex programming and variational inequalities. *SIAM J. Optim.* **7**(4), 951–965 (1997)
60. Tseng, P.: Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.* **109**(3), 475–494 (2001)
61. Tseng, P., Yun, S.: A coordinate gradient descent method for linearly constrained smooth optimization and support vector machines training. *Comput. Optim. Appl.* (2007)
62. Tseng, P., Yun, S.: Block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization. *J. Optim. Theory Appl.* **140**(3), 513–535 (2009). DOI 10.1007/s10957-008-9458-3
63. Tseng, P., Yun, S.: A coordinate gradient descent method for nonsmooth separable minimization. *Math. Program.* **117**(1-2, Ser. B), 387–423 (2009)
64. Vandenberghe, L., Boyd, S.: Semidefinite programming. *SIAM Rev.* **38**(1), 49–95 (1996)
65. Wang, Y., Yang, J., Yin, W., Zhang, Y.: A new alternating minimization algorithm for total variation image reconstruction. *SIAM J. Imaging Sci.* **1**(3), 248–272 (2008)
66. Wen, Z.: First-order methods for semidefinite programming. Ph.D. thesis, Columbia University (2009)
67. Wen, Z., Goldfarb, D., Ma, S., Scheinberg, K.: Row by row methods for semidefinite programming. Tech. rep., Dept of IEOR, Columbia University (2009)
68. Wen, Z., Goldfarb, D., Yin, W.: Alternating direction augmented lagrangian methods for semidefinite programming. *Mathematical Programming Computation* **2**(3-4), 203–230 (2010)
69. Wen, Z., Yin, W., Zhang, Y.: Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. Tech. rep., Rice University (2010). CAAM Technical Report TR10-07
70. Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.): *Handbook of semidefinite programming*. Kluwer Academic Publishers, Boston, MA (2000)
71. Yang, J., Zhang, Y.: Alternating direction algorithms for l1-problems in compressive sensing. Tech. rep., Rice University (2009)
72. Ye, C., Yuan, X.: A descent method for structured monotone variational inequalities. *Optimization Methods and Software* **22**, 329–338 (2007)
73. Ye, J.C., Webb, K.J., Bouman, C.A., Millane, R.P.: Optical diffusion tomography by iterative-coordinate-descent optimization in a bayesian framework. *J. Opt. Soc. Am. A* **16**(10), 2400–2412 (1999)
74. Yu, Z.: Solving semidefinite programming problems via alternating direction methods. *J. Comput. Appl. Math.* **193**(2), 437–445 (2006)
75. Yun, S., Toh, K.C.: A coordinate gradient descent method for l1-regularized convex minimization. *Comput. Optim. Appl.* (2009)
76. Zhang, F. (ed.): The Schur complement and its applications, *Numerical Methods and Algorithms*, vol. 4. Springer-Verlag, New York (2005)
77. Zhang, Y.: User's guide for YALL1: Your algorithms for l1 optimization. Tech. rep., Rice University (2009)
78. Zhao, X., Sun, D., Toh, K.: A Newton-CG augmented lagrangian method for semidefinite programming. *SIAM Journal on Optimization* **20**, 1737–1765 (2010)

# Chapter 20

## Projection Methods in Conic Optimization

Didier Henrion and Jérôme Malick

### 20.1 Introduction, Motivations, Outline

#### 20.1.1 Projection onto Semidefinite Positive Matrices

Consider the space  $\mathcal{S}_n$  of symmetric  $n$ -by- $n$  matrices, equipped with the norm associated to the usual inner product

$$\langle X, Y \rangle = \sum_{i,j=1}^n X_{ij}Y_{ij} = \text{trace}(X^\top Y).$$

The subset  $\mathcal{S}_n^+$  made of positive semidefinite matrices forms a closed convex cone of  $\mathcal{S}_n$ . A general result for closed convex sets yields that we can project onto  $\mathcal{S}_n^+$ : given  $C \in \mathcal{S}_n$ , there exists an unique element of  $\mathcal{S}_n^+$  (called the projection of  $C$  onto  $\mathcal{S}_n^+$  and denoted by  $\text{Proj}_{\mathcal{S}_n^+}(C)$ ) such that

$$\|\text{Proj}_{\mathcal{S}_n^+}(C) - C\| = \min_{X \in \mathcal{S}_n^+} \|X - C\|.$$

It turns out that we also have an explicit expression of this projection, through the spectral decomposition of  $C$ . Consider indeed the decomposition

$$C = U \text{Diag}(\lambda_1, \dots, \lambda_n) U^\top$$

---

D. Henrion

CNRS, LAAS, Toulouse, 7 avenue du colonel Roche, 31077 Toulouse (France)

Faculty of Electrical Engineering, Czech Technical University in Prague, Technická 2,  
16626 Prague (Czech Republic)  
e-mail: [henrion@laas.fr](mailto:henrion@laas.fr)

J. Malick (✉)

CNRS, LJK, Grenoble, INRIA, 655, avenue de l'Europe, 38334 Saint Ismier (France)  
e-mail: [jerome.malick@inria.fr](mailto:jerome.malick@inria.fr)

where  $\lambda_1 \geq \dots \geq \lambda_n$  are the eigenvalues of  $C$  and  $U$  is a corresponding orthonormal matrix of eigenvectors of  $C$ ; then the projection of  $C$  onto  $S_n^+$  is

$$\text{Proj}_{S_n^+}(C) = U \text{Diag}(\max(0, \lambda_1), \dots, \max(0, \lambda_n)) U^\top. \quad (20.1)$$

This result was noticed early by statisticians [67] (see also [31]), and since then this projection has been widely used. We notice that this result generalizes nicely for “spectral sets”; see [47]. Note also that the numerical cost of computing this projection is essentially that of computing the spectral decomposition of  $C$ , the matrix to project.

The developments of this chapter show that more sophisticated projections onto subsets of  $S_n^+$  are also computable using standard tools of numerical optimization. More specifically, the subsets that we consider are intersections of the cone  $S_n^+$  with a polyhedron (defined as affine equalities and inequalities). Though the projection onto those intersections is not explicit anymore, we still have efficient algorithms to compute them, even for large-scale problems.

### 20.1.2 Projection onto Correlation Matrices

The most famous example of such projections is the projection onto the set of correlation matrices (that are the real symmetric positive semidefinite matrices with ones on the diagonal). It is common to be faced with a matrix that is supposed to be a correlation matrix but for a variety of reasons is not. For example, estimating correlation matrices when data come from different time frequencies may lead to a non-positive semidefinite matrix. Another example is stress-testing in finance: a practitioner may wish to explore the effect on a portfolio of assigning certain correlations differently from the historical estimates, but this operation can destroy the semidefiniteness of the matrix.

These important practical questions have led to much interest in the problem of computing the nearest correlation matrix to a given a matrix  $C$  (see e.g. [32, 50, 63] and [9]). This problem is simply formulated as the projection of  $C$  onto correlation matrices

$$\begin{cases} \min & \frac{1}{2} \|X - C\|^2 \\ & X_{ii} = 1, \quad i = 1, \dots, n \\ & X \succeq 0. \end{cases} \quad (20.2)$$

The methods reviewed in this chapter apply to solving this problem in particular. The point is that this problem (and variants of it) can now be solved efficiently (for sizes up to  $n = 5,000$ ; the only limitation on a standard computer is the memory constraints).

### 20.1.3 Conic Projection Problem

The general problem that we first focus on in this chapter is the following. In the space  $\mathbb{R}^n$  equipped with the standard inner product, we want to compute the projection of a point  $c \in \mathbb{R}^n$  onto the intersection  $\mathcal{K} \cap \mathcal{P}$  where:

- $\mathcal{K}$  is a closed convex cone in  $\mathbb{R}^n$  (that we further assume to have full dimension in  $\mathbb{R}^n$ ; that is,  $\text{int}\mathcal{K} \neq \emptyset$ ).
- $\mathcal{P}$  is a convex polyhedron defined by affine (in)equalities

$$\mathcal{P} := \left\{ x \in \mathbb{R}^n : a_i^\top x = (\text{or } \leq) b_i, \quad i = 1, \dots, m \right\}.$$

We suppose moreover that the intersection  $\mathcal{K} \cap \mathcal{P}$  is nonempty, so that the projection onto the closed convex set  $\mathcal{K} \cap \mathcal{P}$  exists and is unique (see e.g. [35]).

The fact that  $\mathcal{P}$  is defined by both equalities and inequalities does not really matter in our developments. To simplify presentation, one may take only equalities, so that  $\mathcal{P}$  is an affine subspace. We prefer to keep the above loose notation with both equalities and inequalities, because it is closer to projection problems arising in practice, and because it does not impact the basics of projection algorithms. Adding positive slack variables for the inequalities allows us to reformulate the problem as a projection onto the intersection of an affine space with a cone of the form  $\mathcal{K} \times (\mathbb{R}_+)^{m_1}$ .

We note that in general one can project onto a polyhedron  $\mathcal{P}$ . For the case when there are only (independent) equalities in the definition (i.e. if  $\mathcal{P} = \mathcal{A}$  is an affine subspace of the equation  $Ax = b$  with a full-rank matrix  $A$ ), we have the explicit expression of the projection of  $x$

$$\text{Proj}_{\mathcal{A}}(x) = x - A^\top [AA^\top]^{-1}(Ax - b). \quad (20.3)$$

For a general polyhedron  $\mathcal{P}$ , we still can compute the projection  $\text{Proj}_{\mathcal{P}}(x)$  efficiently using quadratic programming solvers (see e.g. [58] or [8]). In this chapter, we make the practical assumption that it is also easy to project onto  $\mathcal{K}$ . Recall from above that we have an easy-to-compute expression of the projection for  $\mathcal{K} = \mathcal{S}_n^+$ ; it turns out to be also the case for the second-order cone (or Lorentz cone)

$$\mathcal{L}_n := \left\{ x \in \mathbb{R}^n : \|(x_1, \dots, x_{n-1})\| \leq x_n \right\}.$$

Though it is easy to project onto  $\mathcal{P}$  and also onto  $\mathcal{K}$  by assumption, the projection onto the intersection  $\mathcal{P} \cap \mathcal{K}$  can still be challenging. The difficulty comes from the presence of both (affine and conic) constraints at the same time. We will see in Sect. 20.2 that many numerical methods to compute the projection onto the intersection use combinations of projections onto  $\mathcal{P}$  and  $\mathcal{K}$  separately.

The geometrical projection problem has an obvious analytical formulation as a least-squares problem, namely minimizing the (squared) norm subject to the conic constraints and the affine constraints:

$$\begin{cases} \min & \frac{1}{2}\|x - c\|^2 \\ x \in \mathcal{P} \cap \mathcal{K}. \end{cases} \quad (20.4)$$

For example, an important subclass of such problems are semidefinite least-squares problems (i.e. when  $\mathcal{K} = \mathcal{S}_n^+$ ):

$$\begin{cases} \min & \frac{1}{2}\|X - C\|^2 \\ \langle A_i, X \rangle = (\text{or } \leq) b_i, & i = 1, \dots, m \\ X \geq 0, \end{cases} \quad (20.5)$$

for  $C, A_i \in \mathcal{S}_n$ . For example, the nearest correlation matrix problem (20.2) is an instance of this later class. Notice finally that problems (20.4) and (20.5) coincide formally with  $x \in \mathbb{R}^n$  collecting the rows of  $X \in \mathcal{S}_n$ . This also explains the slight abuse of notation when writing  $\mathcal{K} = \mathcal{S}_n^+$ . We use this ambiguity  $x \leftrightarrow X$  in particular in Sect. 20.4 to ease presentation of relaxations of polynomial optimization problems.

### 20.1.4 Linear Conic Optimization Problem

The second development of this chapter is about a more standard topic: solving linear conic optimization problems. With the above notation, these problems can be expressed as

$$\begin{cases} \min & c^\top x \\ x \in \mathcal{P} \cap \mathcal{K}. \end{cases} \quad (20.6)$$

As presented in this handbook and as well as in the first handbook [70], linear conic programming has been a very active field of research spurred by many applications and by the development of efficient methods.

In this chapter, we explain how conic projections can be used to develop a new family of algorithms for solving linear conic problems. In fact, the projection problem (20.4) can be written as a linear problem of the form (20.6) and then can be solved using usual conic programming solvers (we come back to this at the beginning of Sect. 20.2 and we explain why it is not a good idea to do so). However we will also show the other way around: Sect. 20.3.1 explains that one can also solve the linear conic problem (20.6) by solving projection problems (20.4), more precisely with a succession of (truncated) projection-like problems. So-called regularization methods are presented, discussed and illustrated on solving semidefinite relaxations of combinatorial optimization and polynomial optimization problems having many constraints.

### 20.1.5 Polynomial Optimization

Over the last decade, semidefinite programming has been used in polynomial optimization, namely for deciding whether a multivariate real polynomial is non-negative, or, more generally, to minimize a polynomial on a semialgebraic set (a set described by polynomial inequalities and equations). A hierarchy of embedded linear semidefinite relaxations (of the form (20.6)) can be constructed to generate a monotone sequence of bounds on the global minimum of a polynomial optimization problem. Asymptotic convergence of the sequence to the global minimum can be guaranteed under mild assumptions, and numerical linear algebra can be used to detect global optimality and extract global minimizers. The theory is surveyed in [44] and [43]; the potential applications are numerous (see e.g. in control theory [30] or signal processing [23]).

Section 20.4 reports numerical experiments showing that regularization algorithms based on projections outperform classical primal-dual interior-point algorithms for solving semidefinite relaxations arising when deciding whether a polynomial is nonnegative, and for globally minimizing a polynomial.

### 20.1.6 Objectives and Outline of This Chapter

This chapter focuses on projection problems that have a simple geometric appeal as well as important applications in engineering. We give references to some of these applications, and we give emphasis on polynomial optimization.

The first goal of this chapter is to sketch the different approaches to solve (20.4). Section 20.2 is devoted to this review, with an emphasis on dual methods (in Sect. 20.2.2). The bottomline is that as soon as we can easily project onto  $\mathcal{K}$  (we have in mind  $\mathcal{L}_n$  and  $\mathcal{S}_n^+$  as well as direct products of these), we have efficient algorithms to project onto the intersection  $\mathcal{K} \cap \mathcal{P}$ .

The second goal of this chapter is to explain how to use these conic projections to build a family of “regularization” methods for linear conic programming. The approach uses standard optimization techniques (proximal algorithms and augmented Lagrangian methods) and has been recently developed for the case  $\mathcal{K} = \mathcal{S}_n^+$ . Section 20.3 presents it in a general framework and underlines the role of conic projections. The final section presents some numerical experiments with regularization methods on polynomial optimization problems, showing the interest of the approach in that context.

This chapter is meant to be an elementary presentation of parts of the material of several papers; among those, our main references are [34, 50, 52, 63, 75] and [56]. We aim at clarifying the ideas, presenting them in a general framework, unifying notation, and most of all, pointing out what makes things work. To this purpose, we have to elide some technical points; in particular, we discuss algorithms, but we do not give convergence results. We try to give precise references throughout the text on these lacking points.

## 20.2 Conic Projections: Algorithms and Applications

This section reviews the methods for solving the conic projection problem (20.4), presenting them in chronological order. We sketch the main ideas and give references; we do not get into much details. Discussions about convergence issues and numerical comparisons are beyond the scope of this section.

Beside interior-point methods, the basic idea of all the approaches is to somehow separate the two constraint-sets  $\mathcal{K}$  and  $\mathcal{P}$  and to use the projections onto them successively: this is obvious for alternating projections and alternating directions methods; it is also the case for dual methods (we focus on this latter method in Sect. 20.2.2). The point is that we can solve the conic projection problem (20.4) efficiently (by dual algorithms in particular).

To simplify presentation, we stick here with the projection problem (20.4), but the approach generalizes in two directions. First, we could replace the cone  $\mathcal{K}$  by any closed convex set: in this more general case, the developments are similar, with slightly more complicated expressions of dual objects (a related work is [54]). Second, we could consider problems with strongly convex quadratic objective functions, such as

$$\left\{ \begin{array}{ll} \min & (x - c)^\top Q(x - c) + d^\top x \\ x \in & \mathcal{K} \cap \mathcal{P} \end{array} \right. \quad (20.7)$$

with  $Q$  positive definite. Such problems can be phrased as projection problems with respect to  $\|x\|_Q = \sqrt{x^\top Q x}$  the norm associated to  $Q$ . The practical technical assumption is then that one can project onto  $\mathcal{K}$  with respect to  $\|\cdot\|_Q$  (which is not easy in general).

### 20.2.1 Computing Conic Projections

#### 20.2.1.1 Using Linear Conic Programming

A tempting method to solve (20.4) is to cast this projection problem as a usual linear conic programming problem, so that we can use the powerful tools developed for this case. There are several ways to do so; a simple one consists in pushing down the objective function with an additional variable  $t$ : (20.4) is indeed equivalent to linear conic program

$$\left\{ \begin{array}{ll} \min & t \\ x \in & \mathcal{P} \\ x - c = & z \\ (x, (z, t)) \in & \mathcal{K} \times \mathcal{L}_{n+1} \end{array} \right.$$

where the variable  $z \in \mathbb{R}^n$  is then introduced to express the additional second-order conic constraint appearing in the constraints. This problem can be readily given

to usual conic solvers, for example interior-points methods, like SeDuMi [69] or SDPT3 [73] under Matlab. Unfortunately, adding  $(z, t)$  makes the computational cost and memory space needed by a standard primal-dual interior-point method increase, and numerical testing confirms that the method is not viable in general (as mentioned e.g. in [32, 72]).

We note furthermore that the projection problem (20.4) is a quadratic conic programming problem, hence a special case of nonlinear conic optimization problems. We could solve (20.4) by algorithms and software devoted to nonlinear conic optimization problems such as the penalization method of [40]. However those methods would not use the special structure of (20.4), and as the above approach by linear conic programming, they would be efficient only for small-size projection problems. The projection problems are important enough to design algorithms specifically to them, as presented in the sequel. Note that we are not aware of a tailored penalization algorithm for (20.4).

### 20.2.1.2 Alternating Projections

The alternating projection method is an intuitive algorithmic scheme to find a point in the intersection of two sets: it consists in projecting the initial point onto the first set, then projecting the new point onto the second set, and then projecting again the new point onto the first and keep on projecting alternatively. In other words, it consists in repeating:

$$\begin{cases} x_{k+1} = \text{Proj}_{\mathcal{K}}(y_k) \\ y_{k+1} = \text{Proj}_{\mathcal{P}}(x_{k+1}) \end{cases} \quad (20.8)$$

If the two sets have a “regular” intersection, this algorithm converges linearly to a point in  $\mathcal{P} \cap \mathcal{K}$  and we know the speed of convergence (for two convex sets, see e.g. [20]; for the general case, see the local result of [46]).

We can modify this simple alternating projection scheme by adding a correction step (called Dykstra’s correction [24]) at each iteration (20.8)

$$\begin{cases} x_{k+1} = \text{Proj}_{\mathcal{K}}(z_k) \\ y_{k+1} = \text{Proj}_{\mathcal{P}}(x_{k+1}) \\ z_{k+1} = z_k - (x_{k+1} - y_{k+1}). \end{cases} \quad (20.9)$$

This modification ensures the convergence of the sequence  $(x_k)_k$  to the projection  $\text{Proj}_{\mathcal{K} \cap \mathcal{P}}(c)$  – and not only to a point in the intersection  $\mathcal{K} \cap \mathcal{P}$ . This approach was proposed by [32] for the nearest correlation matrix problem (20.2). It generalizes to (20.4) since it is easy to project onto  $\mathcal{P}$  and we assume that it is the same for  $\mathcal{K}$ . We will see that dual methods and alternating direction methods can be interpreted as variants of this basic geometrical method.

### 20.2.1.3 Dual Methods

The conic programming problem (20.4) looks more complicated than a usual conic programming problem with linear function instead of a norm as objective function. It turns out that the strong convexity of the objective function provides nice properties to the dual problem that can then be solved efficiently.

The dual approach was proposed for the conic least-squares problem (20.4) in [50], later revisited by [16] for the case of  $\mathcal{K} = \mathcal{S}_n^+$ , and then enhanced by [63] and [9] for the projection onto correlation matrices. In the next section, we give more details and more references about this approach.

### 20.2.1.4 Interior Points

As a convex optimization problem, (20.4) can be attacked with the interior-point machinery [57], assuming that both the cone  $\mathcal{K}$  and its polar cone

$$\mathcal{K}^\circ := \left\{ s \in \mathbb{R}^n : s^\top x \leq 0 \text{ for all } x \in \mathcal{K} \right\}$$

are equipped with so-called self-concordant barriers (as is the case for  $\mathcal{L}_n, \mathcal{S}_n^+$ ). The approach consists in solving perturbed optimality conditions of (20.4). As any projection problem, notice that the optimality condition is

$$\bar{x} \in \mathcal{P} \cap \mathcal{K}, \quad (c - \bar{x})^\top (x - \bar{x}) \leq 0, \quad \text{for all } x \in \mathcal{P} \cap \mathcal{K}.$$

To write down the optimality conditions more concretely, let us make explicit the affine constraints with the help of  $A_E \in \mathbb{R}^{n \times m_E}$  and  $A_I \in \mathbb{R}^{n \times m_I}$  as

$$\begin{cases} \min & \|x - c\|^2 \\ & A_E x = b_E, A_I x \leq b_I \\ & x \in \mathcal{K}. \end{cases} \quad (20.10)$$

Under a non-degeneracy assumption (e.g. Slater condition, see next section), the optimality conditions of (20.10) give the complementarity system

$$\begin{cases} x - c + u + A_E^\top y + A_I^\top z = 0 \\ A_E x = b_E, y \in \mathbb{R}^{m_E} \\ A_I x \leq b_I, z \in \mathbb{R}_{+}^{m_I}, z^\top (A_I x - b_I) = 0 \\ x \in \mathcal{K}, u \in \mathcal{K}^\circ, u^\top x = 0. \end{cases}$$

Roughly speaking, an interior-point approach consists in perturbing the complementary equations above and keeping other equations satisfied. (We will see

that the forthcoming dual approach goes exactly the other way around.) A first interior-point method is proposed in [3] for the nearest correlation matrix problem (20.2). Interior-point methods for general quadratic SDP are introduced and tested on projection problems (20.4) in [74] and [72].

### 20.2.1.5 Alternating Directions

The alternating direction method is a standard method in variational analysis (see e.g. [26]), going back to [21]. This method was proposed by [37] for solving the semidefinite projection problem (20.5) and by [71] for more general quadratically constrained quadratic SDP. The idea of the method is to exploit the separable structure of the problem, as follows. Let us duplicate the variables to write the equivalent problem

$$\begin{cases} \min & \frac{1}{2}\|x - c\|^2 + \frac{1}{2}\|y - c\|^2 \\ & x = y \\ & x \in \mathcal{K}, \quad y \in \mathcal{P}. \end{cases} \quad (20.11)$$

The alternating direction method applied to (20.11) gives the following scheme: consider the augmented Lagrangian function

$$L(x, y; z) = \frac{1}{2}\|x - c\|^2 + \frac{1}{2}\|y - c\|^2 - \langle z, x - y \rangle + \frac{\beta}{2}\|x - y\|^2;$$

the minimization of  $L$  with respect to primal variables  $(x, y)$  is decomposed in two steps, so that an augmented Lagrangian iteration is

$$\begin{cases} x_{k+1} = \operatorname{argmin}_{x \in \mathcal{K}} L(x, y_k, z_k) \\ y_{k+1} = \operatorname{argmin}_{y \in \mathcal{P}} L(x_{k+1}, y, z_k) \\ z_{k+1} = z_k - \beta(x_{k+1} - y_{k+1}). \end{cases}$$

It is not difficult to prove that the two above minimizations boil down to projections, more specifically

$$x_{k+1} = \operatorname{Proj}_{\mathcal{K}}\left(\frac{\beta y_k + z_k + c}{1 + \beta}\right), \quad y_{k+1} = \operatorname{Proj}_{\mathcal{P}}\left(\frac{\beta x_{k+1} - z_k + c}{1 + \beta}\right).$$

Thus the approach alternates projections onto  $\mathcal{P}$  and  $\mathcal{K}$  to compute the projection onto  $\mathcal{K} \cap \mathcal{P}$ ; it can thus be seen as a modification of the simple alternating projection scheme (20.8), with the same flavour as Dykstra modification (20.9).

## 20.2.2 More on Dual Approach

### 20.2.2.1 Apply Standard Machinery

Let us give more details about the dual approach for solving (20.4). Following [50], we apply the standard mechanism of Lagrangian duality to this problem; we refer to [35, Chap. XII] and [14, Chap. 5] for more on this mechanism in general.

Let us consider the more explicit form (20.10), and denote also by  $A := [A_E; A_I]$  and  $b := [b_E; b_I]$  the concatenation of the affine constraints. We dualize affine constraints only: introduce the Lagrangian, a function of primal variable  $x \in \mathcal{K}$  and dual variable  $(y, z) \in \mathbb{R}^{m_E} \times \mathbb{R}_+^{m_I}$

$$L(x; y, z) := \frac{1}{2} \|c - x\|^2 - y^\top (A_E x - b_E) - z^\top (A_I x - b_I), \quad (20.12)$$

and the corresponding concave dual function

$$\theta(y, z) := \min_{x \in \mathcal{K}} L(x; y, z), \quad (20.13)$$

which is to be maximized. There is no more affine constraint in the above minimum, and it is easy to prove ([50, Theorem 3.1]) that the problem corresponds to a projection onto  $\mathcal{K}$ : there exists a unique point which reaches the above minimum, namely

$$x(y, z) := \text{Proj}_{\mathcal{K}}(c + A_E^\top y + A_I^\top z), \quad (20.14)$$

so we have

$$\theta(y, z) = b_E^\top y + b_I^\top z + \frac{1}{2} (\|c\|^2 - \|x(y, z)\|^2). \quad (20.15)$$

It is also not difficult to show [50, Theorem 3.2] that the concave function  $\theta$  is differentiable on  $\mathbb{R}^m$ , and that its gradient

$$\nabla \theta(y, z) = -Ax(y, z) + b \quad (20.16)$$

is Lipschitz continuous. As any function with Lipschitz gradient,  $\theta$  is twice differentiable almost everywhere, but not everywhere (this basically relies on the differentiability properties of  $\text{Proj}_{\mathcal{K}}$ ; for the case  $\mathcal{K} = \mathcal{S}_n^+$ , see more in [68] and [53] among others).

The dual problem is thus

$$\max_{(y, z) \in \mathbb{R}^{m_E} \times \mathbb{R}_+^{m_I}} \theta(y, z) \quad (20.17)$$

Strong duality (the optimal values of (20.10) and (20.17) coincide) holds under a standard assumption in convex optimization. The so-called (weak) Slater assumption (see e.g. [6, 35]) is in our context:

$$\exists \bar{x} \in \mathcal{P} \cap \text{int } \mathcal{K}. \quad (20.18)$$

In fact, this assumption yields moreover that there exists solutions to (20.17) (note that the assumption has also a natural geometrical appeal in context of projection methods, see [34, Sect. 3]). Finally we get directly the projection from dual solutions: let  $(y^*, z^*)$  be a (dual) solution of (20.17), the (primal) solution  $x^*$  of (20.4) is the associated  $x^* = x(y^*, z^*)$  (see [50, Theorem 4.1]).

### 20.2.2.2 Apply Standard Algorithms

To compute the projection of  $c$  onto  $\mathcal{P} \cap \mathcal{K}$ , we just have to solve the dual problem (20.17). Let us have a closer look to this problem: the constraints are simple positivity constraints on the variable corresponding to the dualization of inequality constraints; the dual function is a differentiable concave function with Lipschitz gradient. This regularity has a huge impact in practice: it opens the way for using standard algorithms for nonlinear optimization. Hence we can use any of the following numerical methods to solve (20.17) (as soon as the software can deal with the constraints  $z_i \geq 0$ ):

1. Gradient methods: standard methods [6] or more evolved ones, as e.g. Nesterov's method [55]
2. Newton-like methods: quasi-Newton, limited memory quasi-Newton, inexact Newton, Newton-CG, see textbooks [58] and [8] – with the restriction that  $\theta$  is not twice differentiable everywhere, so that we have to use the so-called semismooth Newton methods, see [62]

For example, [50] uses a quasi-Newton method for solving (20.5), and [63] uses a semismooth inexact Newton method for solving (20.2). We come back on these two methods in the next section to give more practical details.

We also mention here the so-called inexact smoothing method of [27] which consists in writing the optimality conditions of the dual problem (20.17) as a nonsmooth fixed point problem (and solving it by combining smoothing techniques and an inexact Newton method; see e.g. [58]).

The dual problem (20.17) can thus be attacked with classical tools or more evolved techniques. In practice, the choice of the solving method depends on the structure of the problem and the target level of sophistication.

We call dual projection methods any method using an optimization code for functions with Lipschitz gradient to maximize  $\theta$  on  $\mathbb{R}^{m_E} \times \mathbb{R}_+^{m_I}$ . Specifically, a dual projection method generates a maximizing dual sequence  $\{y_k, z_k\}_k$  together with the primal sequence  $x_k = x(y_k, z_k)$  such that:

$$\theta(y_k, z_k) = b_E^\top y_k + b_I^\top z_k + \frac{1}{2}(\|c\|^2 - \|x_k\|^2) \quad (20.19)$$

$$\nabla \theta(y_k, z_k) = -Ax_k + b. \quad (20.20)$$

We notice that in our numerical experiments with dual methods, we have observed better behaviour and convergence when the (strong) Slater assumption holds (that is, when (20.18) holds and moreover  $A_E$  is full rank).

### 20.2.2.3 More Algorithmic Details (for the Case Without Inequalities)

We detail now further some algorithmic issues. To simplify we focus on the case without inequalities ( $m_I = 0$ , no dual variables  $z$ ). Iterations of most algorithms for maximizing  $\theta$  can be written as

$$y_{k+1} = y_k + \tau_k W_k \nabla \theta(y_k). \quad (20.21)$$

Note that the usual stopping test of these methods has an intrinsic meaning: a threshold condition on the gradient

$$\|\nabla \theta(y_k)\| = \|Ax_k - b\| \leq \varepsilon \quad (20.22)$$

controls in fact the primal infeasibility. Among these methods, let us discuss further the three following ones.

#### Gradient Descent with Constant Step-Size

We have a remarkable result: the gradient method in an adapted metric, namely (20.21) with

$$W_k = [AA^\top]^{-1} \quad \text{and} \quad \tau_k = 1, \quad (20.23)$$

corresponds exactly to the alternating projection method (20.9) (see [50] for a proof in the special case of correlation matrices, and [34] for the proof in general). We thus have a (surprising) dual interpretation of the primal projection method. Using descent schemes more evolved than a simple gradient descent (see below) then leads to (dual) projection methods that can be seen as improvements of the basic alternating projection method.

#### BFGS Quasi-Newton Method

The method is known to be very efficient in general, and have many industrial applications (one of the most striking is in weather forecasting [25]). The method can be readily applied to the dual problem, since it requires no more information than (20.19):  $W_k$  is constructed with successive gradients with the BFGS formula and  $\tau_k$  is well-chosen with a Wolfe line-search (see e.g. [8]). The initial paper about dual methods [50] proposes to use this method in general and reports very good numerical results on the nearest correlation matrix problem (20.2). Since then, this dual method has been used successfully to solve real-life projection

problems in numerical finance (among them: the problem of calibrating covariance matrices in robust portfolio selection [50, 5.4]). A simple Matlab implementation has been made publicly available together with [34] for pedagogical and diffusion purposes.

### Generalized (or Semismooth) Newton

A pure Newton method would be to use  $\tau_k = 1$  and  $W_k = [H_k]^{-1}$  with the Hessian  $H_k$  of  $\theta$  at the current iterate  $y_k$ . In practice, an inexact generalized Newton method is used for the following reasons.

As mentioned earlier,  $\theta$  is differentiable but not twice differentiable (though its gradient is Lipschitz continuous). We can still replace the usual Hessian by a matrix  $H_k \in \partial_c^2 \theta(y_k)$  the Clarke generalized Hessian of  $\theta$  at  $y_k$  [18]. Computing a matrix in  $\partial_c^2 \theta(y_k) \subset \mathcal{S}_n^+$  amounts to computing an element of the Clarke generalized Jacobian of the projection onto the cone  $\partial_c \text{Proj}_{\mathcal{K}}$  since we have (see [36])

$$\partial_c^2 \theta(y_k) d = A \partial_c \text{Proj}_{\mathcal{K}}(c + A^\top y_k) A^\top d \quad \text{for all } d \in \mathbb{R}^m$$

We can often compute an element of  $\partial_c \text{Proj}_{\mathcal{K}}$ . For example, we even have an explicit expression of the whole  $\partial_c \text{Proj}_{\mathcal{S}_n^+}$  [53].

For overall efficiency of the method, the Newton direction  $d_k$  is computed by solving the system  $H_k d = \nabla \theta(y_k)$  approximately, usually by conjugate gradient (CG) type methods. More precisely, the idea of so-called Newton-CG (also called inexact Newton going back to [19]) is to stop the inner iteration of CG when

$$\|H_k d + \nabla \theta(\lambda_k)\| \leq \eta_k \|\nabla \theta(\lambda_k)\| \tag{20.24}$$

with small  $\eta_k$  (see e.g. [58]). Note that preconditioning the Newton system is then crucial for practical efficiency. The nice feature of this algorithm is that  $H_k$  has just to be known through products  $H_k d$  so that large-scale problems can be handled. In our context, the main work on this method is [63] about the nearest correlation matrix problem; we come back to it in the next section.

We finish here with a couple of words about convergence of this Newton dual method. In general (see [62]), the two conditions to prove local superlinear convergence are that the minimum is strong (i.e. all elements of the generalized Hessian are positive definite), and the function has some smoothness (namely, the so-called semismoothness). In our situation, the two ingredients implying those conditions are the following ones:

- The intersection has some “nondegeneracy”, in the sense of [13, 4.172] and [2, Definition 5]. This allows us to prove  $\partial_c^2 \theta(y_k) > 0$  (see e.g. [63] for a special case).
- The convex cone  $\mathcal{K}$  has some “regularity”. An example of sufficient regularity is that  $\mathcal{K}$  is a semialgebraic set (i.e. defined by a finite number of polynomial

(in)equalities). Indeed for semialgebraic convex sets, the projection  $\text{Proj}_{\mathcal{K}}$  and then  $\theta$  are automatically semismooth [5] (which is the property needed to apply the convergence results of [62]). This is the case for direct products of the cones  $\mathcal{L}_n$  and  $\mathcal{S}_n^+$  (for which we even have strong semismoothness [68] so in fact quadratic convergence).

#### 20.2.2.4 Illustration on Nearest Correlation Matrix Problem

We give a rough idea of the efficiency of the dual approach on the projection problem (20.2). The first numerical results of [50, Sect. 4] show that the dual approach copes with large-scale problems, in reporting that one can solve in a couple of minutes projection problems of size around one thousand. By using the dual generalized Newton method (instead of quasi-Newton as in [50]), the algorithm of [63], improved later by [9], gives very nice results in both practice and theory. Nondegeneracy of the constraints and then of the generalized Hessian is proved in [63, Proposition 3.6]: as recalled above, this technical point leads to quadratic convergence of the method [63, Proposition 5.3].

Today's state of the art is that one can solve nearest correlation matrix problems of big size (say, up to 4,000–5,000) in a reasonable amount of computing time (say, less than 10 min on a standard personal computer). The only limitation seems to be the memory constraint to store and deal with dense large-scale data.

To give a more precise idea, let us report a couple of results from [9]. The implementation of their dual algorithm is in Matlab with some external Fortran subroutines (for eigenvalues decomposition in particular). The stopping criterion is set to

$$\|\nabla\theta(y_k)\| \leq 10^{-7}n. \quad (20.25)$$

We consider the nearest correlation matrix problems for two (non-SDP) matrices with unit diagonal (of size  $n_1 = 1,399$  and  $n_2 = 3,120$ ) provided by a fund management company. The dual method solves them in around 2 and 15 min., respectively, on a very standard machine (see more details in [9]).

We finish with a last remark about accuracy. The approximate correlation matrix  $X$  that is computed by such a dual method is often just what is needed in practice. It might happen though that a special application requires a perfect correlation matrix – that is, with exactly ones on the diagonal, whereas  $X$  satisfies only (by (20.25))

$$\left( \sum_{i=1}^n (X_{ii} - 1)^2 \right)^{-1/2} \leq 10^{-7}n.$$

A simple post-treatment corrects this. Setting diagonal elements to ones may destroys the positiveness, so we apply the usual transformation that computes the associated correlation matrix  $\bar{X}$  from a covariance matrix  $X$ , namely

$$\bar{X} = D^{-1/2} X D^{-1/2} \quad \text{and} \quad D = \text{diag}(X).$$

This operation increases the distance from  $C$ ; but the error is still under control (by  $\varepsilon/(1-\varepsilon)$ ; see [9, Proposition 3.2]).

### 20.2.3 Discussion: Applications, Generalizations

#### 20.2.3.1 Direct or Indirect Applications

Conic projection problems with the positive semidefinite cone (like  $\mathcal{K} = \mathcal{S}_n^+$ ,  $\mathcal{K} = \mathcal{S}_n^+ \times (\mathbb{R}^+)^p$  or  $\mathcal{K} = \mathcal{S}_{n_1}^+ \times \cdots \times \mathcal{S}_{n_p}^+$ ) are numerous in engineering. Constructing structured semidefinite matrices, for example, are naturally modeled this way. Such problems naturally appear in finance for constructing structured covariance matrices (as a calibration step before simulations); they also appear in many other fields, such as in control (e.g. [45]), in numerical algebra (e.g. [1]), or in optics (e.g. [59]), to name a few of them.

Conic projections also appear as inner subproblems within more involved optimization problems. Solving efficiently these inner problems is often the key to numerical efficiency of the overall approach. Let us give some examples.

- *Linear conic programming.* So-called regularization methods for solving (20.6) use the conic projection problem as an inner subproblem; these methods are studied in Sect. 20.3.
- *Weighted projections.* For given weights  $H_{ij} \geq 0$ , consider the semidefinite projection (20.5) with a different objective function

$$\begin{cases} \min & \frac{1}{2} \sum_{i,j=1}^n H_{ij}(X_{ij} - C_{ij})^2 \\ & \langle A_i, X \rangle = (\text{or } \leq) b_i, \quad i = 1, \dots, m \\ & X \geq 0. \end{cases}$$

An augmented Lagrangian approach for this problem [64] produces a projection-like inner problem, which is solved by a semismooth Newton method (recall the discussion of the previous section).

- *Low-rank projections.* Consider the semidefinite projection problem (20.5) with additional rank-constraint

$$\begin{cases} \min & \frac{1}{2} \|X - C\|^2 \\ & \langle A_i, X \rangle = (\text{or } \leq) b_i, \quad i = 1, \dots, m \\ & X \geq 0, \quad \text{rank } X = r. \end{cases} \quad (20.26)$$

This difficult non-convex calibration problem has several applications in finance and insurance industry (e.g. pricing interest rate derivatives for some models; see e.g. [11]). Two approaches (by augmented Lagrangian [49] and by penalty techniques [28]) have been recently proposed to solve these types of problems; both approaches solve a sequence of projection-like subproblems. The numerical engine is a dual semismooth truncated Newton algorithm for computing projections.

For these applications of conic projections, the techniques and the arguments are often the same, but are redeveloped for each particular projection problem encountered. We hope that the unified view of Sect. 20.2 can bring forth the common ground of these methods and to better understand how and why they work well. We finish this section by pointing out an easy geometrical application.

### 20.2.3.2 Application for Solving Conic Feasibility Problems

The conic feasibility problem consists simply in finding a point  $x$  in the intersection  $\mathcal{K} \cap \mathcal{P}$ . Many engineering problems can be formulated as semidefinite or conic feasibility problems (for example in robust control [7] where an element in the intersection is a certificate of stability of solutions of differential equations). Section 20.4.2 focuses on semidefinite feasibility problems arising when testing positivity of polynomials. We refer to the introduction of [34] for more examples and references.

A simple and natural technique for solving conic feasibility problems is just to project a (well-chosen) point onto the intersection  $\mathcal{K} \cap \mathcal{P}$  (by dual projection methods for example). In [34], a comparative study of such a conic projection method with the usual approach using SeDuMi was carried out precisely on polynomial problems. It was shown there that an elementary Matlab implementation can be competitive with a sophisticated primal-dual interior-point implementation. This would even have a better performance if an initial heuristic for finding a good point to project could be determined (the numerical experiments of [34, Sect. 6] simply use  $c = 0$ ). An answer to this latter point is provided by the regularization methods of the next section.

## 20.3 Projections in Regularization Methods

We focus in this section on standard linear conic programming. We show that, following classical convex optimization techniques, conic projections can be used to solve linear conic programming problems.

There exist many numerical methods for solving linear conic problem (20.6) (see the first handbook [70]). But on the other hand, there also exist big conic problems, and especially big SDP problems, that make all the standard methods fail. Relaxations of combinatorial optimization problems and polynomial optimization problems yield indeed challenging problems. This motivates the development of new algorithmic schemes.

The strategy that we present in this section exploits the efficiency of projection methods by developing proximal algorithms for linear conic programming. We generalize the developments of [52], and give all way long references to related works. As for numerical aspects, the target problems are semidefinite programs with the number of constraints possibly very large (more than 100,000).

### 20.3.1 Proximal Method for Linear Conic Programming

#### 20.3.1.1 Apply Classical Techniques of Convex Optimization

The proximal algorithm is a classical method of convex optimization and variational analysis: it goes back from the 1970s with premises in [10], the first work [51] and the important reference [66]. The driving idea of the proximal algorithm is to add quadratic terms to the objective function to “regularize” the problem (ensuring existence, uniqueness, and stability of solutions). A (primal) proximal method of the linear conic problem (20.6) goes along the following lines.

Consider the problem with respect to  $(x, p)$

$$\begin{cases} \min & c^\top x + \frac{1}{2t} \|x - p\|^2 \\ p \in \mathbb{R}^n, & x \in \mathcal{P} \cap \mathcal{K}. \end{cases}$$

By minimizing first with respect to  $p$ , we see that this problem is equivalent to the primal linear conic problem (20.6). We have added to the objective function a quadratic “regularizing” term  $\|x - p\|^2$  with the so-called “prox-parameter”  $t$ . The idea now is to solve this problem in two steps: first with respect to  $x$ , and second to  $p$ :

$$\begin{cases} \min & \left( \min_{x \in \mathcal{P} \cap \mathcal{K}} c^\top x + \frac{1}{2t} \|x - p\|^2 \right) \\ p \in \mathbb{R}^n & \end{cases}. \quad (20.27)$$

The outer problem is thus the minimization with respect to  $p$  of the function

$$F(p) := \begin{cases} \min & c^\top x + \frac{1}{2t} \|x - p\|^2 \\ x \in \mathcal{P} \cap \mathcal{K} & \end{cases} \quad (20.28)$$

which is the result of the inner optimization problem parameterized by  $p$ . As such defined,  $F$  is the so-called Moreau–Yosida regularization of the function  $x \rightarrow c^\top x + i_{\mathcal{P} \cap \mathcal{K}}(x)$  the linear objective function plus the indicator function of the intersection (see e.g. [35, Chap. XV.4]).

The connection with the previous developments of this chapter is then obvious: the above inner problem is essentially a projection problem as studied in the previous section (see (20.7)). The solution of the inner problem (the “projection”) is called the proximal point and denoted

$$\text{Prox}(p) := \begin{cases} \operatorname{argmin}_{x \in \mathcal{P} \cap \mathcal{K}} & c^\top x + \frac{1}{2t} \|x - p\|^2 \end{cases}$$

Note that, for simplicity, the dependence of  $F$  and  $\text{Prox}$  with respect to  $t$  is dropped in notation.

### 20.3.1.2 Primal Proximal Algorithm for Conic Programming

Applying basic convex analysis properties, it is easy to prove (see e.g. [35, Chap. XV.4]) that the Moreau–Yosida regularization  $F$  is convex and differentiable with gradient  $\nabla F(p) = (p - \text{Prox}(p))/t$ . The optimality condition of the unconstrained minimization of  $F$  is then simply

$$\bar{p} = \text{Prox}(\bar{p}). \quad (20.29)$$

Moreover a fixed-point  $\bar{p}$  of the Prox operator is also a solution of the initial linear conic problem (20.6): observe indeed that  $\bar{p}$  is feasible and reaches the optimal value, since

$$\text{val}(20.6) = \min F(p) = F(\bar{p}) = c^\top \bar{p}. \quad (20.30)$$

A (primal) proximal algorithm for solving (20.6) then consists of a fixed-point algorithm on (20.29)

$$p_{k+1} = \text{Prox}(p_k). \quad (20.31)$$

Since computing  $\text{Prox}(p_k)$  corresponds to solving a projection problem, we can use any of the algorithmic schemes described in Sect. 20.2.1 to implement (20.31) inside of this proximal algorithm. We call the proximal method the outer algorithm, and the chosen projection algorithm, the inner algorithm. We study in Sect. 20.3 the family of proximal algorithms obtained when dual projection algorithms of Sect. 20.2.2 are used as inner algorithms.

As we have an iterative optimization algorithm (inner algorithm) inside of another iterative algorithm (outer algorithm), the question of the stopping tests of the inner algorithm is obviously crucial. For practical efficiency, the inner stopping test should somehow depend on some outer information; we come back later in detail to this important point.

So in fact the iteration (20.31) is not carried out exactly, and replaced instead by a looser implementable relation

$$\|p_{k+1} - \text{Prox}(p_k)\| \leq \varepsilon_k. \quad (20.32)$$

Whatever is the inner projection algorithm, we have the general global convergence of the method under the assumption that the sequence of errors  $\varepsilon_k$  goes rapidly to zero.

**Proposition 20.1 (Global convergence).** *Assume that there exist a solution to (20.6). If  $(t_k)_k$  is bounded away from 0 and if the primal proximal algorithm generates a sequence  $(p_k)_k$  such that*

$$\sum_k \varepsilon_k < +\infty \quad (20.33)$$

*then  $(p_k)_k$  converges to a solution  $\bar{p}$  of (20.6).*

*Proof.* The result is straightforward from the general convergence result of proximal algorithms. As a consequence of (20.33) and the existence of a solution to (20.6), the sequence  $(p_k)_k$  is bounded and we can apply [66, Theorem 1]:  $(p_k)_k$  converges to a fixed-point to Prox which is a solution of (20.6) by (20.30).  $\square$

### 20.3.1.3 Dual Point of View: Augmented Lagrangian

We give here some details about the dual interpretation of the above primal algorithmic approach. It is known indeed that a proximal method for a problem corresponds exactly to an augmented Lagrangian method on its dual; we detail this for our case. To simplify writing duals, we abandon the general formulation (20.6), and we suppose that there is no affine inequalities (or that there are incorporated with slack variables in  $\mathcal{K}$ ). So we work from now with the standard form of primal and dual linear conic problems

$$\begin{cases} \min & c^\top x \\ Ax = b \\ x \in \mathcal{K} \end{cases} \quad \text{and} \quad \begin{cases} \max & b^\top y \\ A^\top y - u - c = 0 \\ u \in \mathcal{K}^o. \end{cases} \quad (20.34)$$

Augmented Lagrangian methods are important classical regularization techniques in convex optimization (see [61, 65] for important earlier references, and [35, Chap. XII] for the connection with usual Lagrangian duality). In our situation, a dual augmented Lagrangian method goes along the following lines. Introduce the augmented Lagrangian function  $L$  with parameter  $t > 0$ , for the dual problem (20.34):

$$L(y, u; p) := b^\top y - p^\top (A^\top y - u - c) - \frac{t}{2} \|A^\top y - u - c\|^2.$$

Note that this is just the usual Lagrangian for the problem

$$\begin{cases} \max & b^\top y - \frac{t}{2} \|A^\top y - u - c\|^2 \\ A^\top y - u - c = 0, & u \in \mathcal{K}^o, \end{cases} \quad (20.35)$$

that is the dual problem with an additional redundant quadratic term in the objective. The convex (bi)dual function is then defined as

$$\Theta(p) := \max_{y \in \mathbb{R}^m, u \in \mathcal{K}^o} L(y, u; p). \quad (20.36)$$

The bridge between the primal proximal method and the dual augmented Lagrangian is set in the next proposition, formalizing a well-known result.

**Proposition 20.2.** *With notation above, we have  $\Theta(p) = F(p)$  for  $p \in \mathbb{R}^n$ .*

*Proof.* Just apply [35, XII.5.2.3]: the augmented Lagrangian function  $\Theta(p)$  is the Moreau–Yosida of the usual dual function, which is here

$$c^\top p + i_{\{Ax=b\} \cap \mathcal{K}}(p) = \max_{y, u \in \mathcal{K}^o} b^\top y - p^\top (A^\top y - u - c).$$

This is exactly  $F(p)$  defined by (20.28) (in the case when  $\mathcal{P}$  is just the affine subspace of equation  $Ax = b$ ).  $\square$

The primal regularization by proximal approach and the dual augmented Lagrangian regularization thus correspond exactly to the same quadratic regularization process viewed either on the primal problem or on the dual (20.34).

The developments of this section share similar properties with other augmented Lagrangian-type approaches for conic programming, among them: a primal augmented Lagrangian in [15], a primal-dual augmented Lagrangian in [38] and a penalized augmented Lagrangian in [41].

### 20.3.2 Regularization Methods for Linear Conic Programming

In this section we give more details on primal proximal algorithms (or dual augmented Lagrangian algorithms) that use dual projection methods as inner algorithms to carry out the proximal iteration (20.32). This family of algorithms is introduced for the case  $\mathcal{K} = \mathcal{S}_n^+$  in [52]. They are called regularization algorithms (rather than proximal algorithms, which would focus on the primal point of view only); we keep this terminology here. This section is more technical and could be skipped at a first reading.

Regularization algorithms for conic programming specialize on three points:

1. The dual projection algorithm to compute  $\text{Prox}(x_k)$
2. The rule to stop this inner algorithm
3. The rule to update the prox-parameter  $t_k$

The third point is an inherent difficulty of any practical implementation of proximal methods (e.g. bundle methods, see [17]). We are not aware of general techniques to tackle it. So we focus here on the first two points.

#### 20.3.2.1 Dual Projection Methods as Inner Algorithms

We could use any dual projection algorithm of Sect. 20.2.2 to solve

$$\begin{cases} \min & c^\top x + \frac{1}{2t} \|x - p\|^2 \\ & Ax = b, x \in \mathcal{K}. \end{cases} \quad (20.37)$$

Embedded in a proximal scheme, a dual projection algorithm would lead to the forthcoming overall algorithm for solving linear conic problems (20.34).

Note first that equations (20.14) and (20.15) for the projection-like problem (20.37) become respectively

$$x(y) = \text{Proj}_{\mathcal{K}}(p + t(A^\top y - c)) \quad (20.38)$$

$$\theta(y) = b^\top y + \frac{1}{2t}(\|p\|^2 - \|x(y)\|^2). \quad (20.39)$$

---

**Algorithm** Regularization methods.

---

Outer loop on  $k$  stopped when  $\|p_{k+1} - p_k\|$  small:

Inner loop on  $\ell$  stopped when  $\|Ax_\ell - b\|$  small enough:

Compute  $x_\ell = \text{Proj}_{\mathcal{K}}(p_k + t_k(A^\top y_\ell - c))$  and  $g_\ell = b - Ax_\ell$

Update  $y_{\ell+1} = y_\ell + \tau_\ell W_\ell g_\ell$  with appropriate  $\tau_\ell$  and  $W_\ell$

end (inner loop)

Update  $p_{k+1} = x_\ell$  (and  $t_k$ )

end (outer loop)

---

We use the (slightly loose) formulation (20.21) of the iteration of dual projection methods to write a general regularization algorithm. We index the outer iterations by  $k$  and the inner ones by  $\ell$ .

We discuss several points about the above conceptual algorithm.

- *Memory.* An important feature of regularization methods is the rather low memory requirement. The intrinsic operations of the algorithm are basically the projection onto the cone and the multiplications by  $A$  and  $A^\top$ . If the data has some structure, those multiplications could be performed efficiently (without constructing matrices). Moreover for maximizing  $\theta$  (that is, essentially, implementing  $y_{\ell+1} = y_\ell + \tau_\ell W_\ell g_\ell$ ), we could use algorithms of smooth unconstrained optimization adapted to large-scale problems and then requiring low-memory (as limited memory BFGS or Newton-CG, see e.g. [58] and [8]). We come back to this point later when discussing numerical issues. Roughly speaking, the point is the following: the computer memory can be used for storing problem data and the computation does not require much more extra memory.)
- *Inner restarting.* At the outer iteration  $k$ , the inner projection algorithm can be initialized with the best  $y_\ell$  of the previous iteration  $k - 1$ . This has an intuitive appeal, so that in practice,  $\ell$  keeps increasing over the outer iterations. (Note also that the historical information on gradients may be carried out from iteration  $k - 1$  to  $k$  as well.)
- *Dual variable  $u$ .* It is known that for any  $x \in \mathbb{R}^n$ , the projection onto the polar cone  $\text{Proj}_{\mathcal{K}^\circ}(x)$  is given by  $\text{Proj}_{\mathcal{K}}(x) + \text{Proj}_{\mathcal{K}^\circ}(x) = x$  (together with  $\text{Proj}_{\mathcal{K}}(x)^\top \text{Proj}_{\mathcal{K}^\circ}(x) = 0$ , see [35, III.3.2.5]). When computing  $x_\ell$ , we thus get automatically

$$u_\ell = \text{Proj}_{\mathcal{K}^\circ}(p_k + t_k(A^\top y_\ell - c))/t_k$$

and it holds

$$p_k + t_k(A^\top y_\ell - c) = t_k u_\ell + x_\ell. \quad (20.40)$$

- *Dual outer iterates.* At the end of outer iteration  $k$ , we set (with a slight abuse of notation)  $y_{k+1} = y_\ell$  and  $u_{k+1} = u_\ell$  for  $\ell$  the final iteration of inner algorithm. Thus we have a sequence of primal-dual outer iterates  $(p_k, y_k, u_k) \in \mathcal{K} \times \mathbb{R}^n \times \mathcal{K}^\circ$ . Under some technical assumptions, we can prove a convergence result of the same vein as Proposition 20.1: any accumulation point of the sequence  $(p_k, y_k, u_k)$  is a primal-dual solution of (20.10) (see e.g. Theorem 4.5 of [52] for a proof when  $\mathcal{K} = \mathcal{S}_n^+$ ).

- *Outer stopping test.* We have already noticed in (20.22) that the natural stopping test of dual projection algorithms controls primal infeasibility  $\|Ax_\ell - b\|$ . Interpreted as a fixed point iteration (20.31), the natural stopping of the proximal algorithm is  $\|p_{k+1} - p_k\|$ ; it turns out that this can be interpreted as controlling dual infeasibility. Note indeed that (20.40) yields

$$p_k + t_k(A^\top y_k - c) = t_k u_k + p_k$$

and then we have

$$\|p_{k+1} - p_k\| = t_k \|A^\top y_k - u_k - c\|.$$

- *Normal to interior-point methods.* By construction, conic feasibility  $p \in \mathcal{K}$  (and  $u \in \mathcal{K}^o$ ) and complementary  $x^\top u = 0$  are ensured throughout the algorithm, while primal-dual feasibilities are obtained asymptotically. In contrast, recall that basic interior-point methods maintain primal and dual feasibility and the conic feasibility and work to reach complementarity. Note also that, regularization algorithms give solutions that are usually on the boundary of the cone  $\mathcal{K}$ , since the primal iterates are constructed by projections onto  $\mathcal{K}$ . In contrast again, basic interior-points give solutions as inside of the cone as possible. In a sense, regularization methods are then “normal” to interior point methods.
- *Overall stopping test.* We have seen above that the natural outer and inner stopping rules of the regularization algorithm have a practical interpretation as dual and primal infeasibilities. Since complementary and conic feasibility are ensured by construction, the natural stopping test of the overall algorithm is

$$\max \left\{ \|Ap_k - b\|, \|A^\top y_k - u_k - c\| \right\}. \quad (20.41)$$

In practice, one should divide moreover the two infeasibilities by some constant quantities to get homogeneous ratios.

### 20.3.2.2 Stopping Inner Iterations

For which inner iteration  $\ell$  can we set  $p_{k+1} = x_\ell$  to proceed with the outer iteration? Since we have a loop inside of another, the rule to terminate the inner projection algorithm is indeed an important technical point for regularization algorithms. We discuss three strategies to set up inner stopping rules.

#### Solving Approximately the Inner Problem

The usual stopping inner rule in proximal methods is to stop inner iterations when the current inner iterate  $x_\ell$  is close to the proximal point  $\text{Prox}(p_k)$ . Doing this, the regularization algorithm approximates at best the conceptual proximal algorithm (which requires to solve the inner problem exactly), so that we keep convergence properties (as in Proposition 20.1 for instance).

This is the strategy followed by the regularization method of [75] for  $\mathcal{K} = \mathcal{S}_n^+$ . This paper adopts the dual point of view (augmented Lagrangian) and uses semismooth Newton as dual projection algorithm for inner iterations (remember Sect. 20.2.2). The regularization method thus combines the usual stopping strategy and an efficient inner algorithm (supporting large-scale problems); it gives excellent numerical results on various semidefinite programming test-problems (see the last section of [75]). Under nondegeneracy assumptions, we have moreover proofs of global and local convergences of the inner algorithm as well as the overall regularization method.

### Only One Inner Iteration; Interpretation as Saddle-Point

An opposite strategy is to do only one inner iteration per outer iteration. This cautious strategy is motivated by the following remark. Let us come back to the proximal formulation (20.27) of the linear conic problem. Under Slater assumption (20.18), the inner projection problem can be replaced by its dual (recall (20.15), (20.17) and (20.38)), so that the primal and dual conic problems (20.34) have the saddle-point formulation

$$\min_{p \in \mathbb{R}^n} \left( \max_{y \in \mathbb{R}^m} b^\top y - \frac{1}{2t} (\|p\|^2 - \|\text{Proj}_{\mathcal{K}}(p + t(A^\top y - c))\|^2) \right).$$

With this point of view on the process, the choice of inner stopping conditions appears indeed to be crucial, because the inner and outer loops are antagonistic, as the first minimizes and the second maximizes. The idea of the “Arrow–Hurwicz” approach (see, for instance, [4]) is essentially to proceed with gradient-like iterations with respect to each variable successively.

This is the strategy of the simple regularization method presented in [52, Sect. 5.1]. Doing one inner iteration of (20.21) with  $W_k = [AA^\top]$  and  $\tau_k = 1/t_k$  allows to simplify the regularization algorithm to just one loop with

$$p_{k+1} = \text{Proj}_{\mathcal{K}}(p_k + t_k(A^\top y_\ell - c)) \quad (\text{with } u_{k+1} \text{ as by-product}) \quad (20.42)$$

$$y_{k+1} = y_k + [AA^\top]^{-1}(b - Ap_k)/t_k. \quad (20.43)$$

We retrieve Algorithm 5.1 of [52] by using the proof of Proposition 3.4 in there. This simple regularization algorithm has also an interpretation as the so-called boundary-point method (see [60]) and as an alternating direction method (see Chap. 19 of this Handbook).

In practice, it is important to note that  $AA^\top$  and its Cholesky factorization can be computed only once at the beginning of the algorithm. Even if this is an expensive task in general for problems that have many unstructured constraints (so that  $AA^\top$  is big and unstructured), there exists some cases when  $AA^\top$  is very simple, so that solving the system (20.43) is cheap. This is the case in particular when  $AA^\top$  is diagonal,

as for SDP relaxations of max-cut problem, or  $k$ -max-cut problem [29], frequency assignment problems, see [12, (5)], max-stable problem, see more below, and polynomial minimization problems, see forthcoming Sect. 20.4.

### Something In-Between

An attractive option is to find something in-between the previous two extreme strategies. Explicit rules for the management of  $\varepsilon_k$  in (20.32) should be given, and for numerical efficiency they should be given online. Using off-line rules independent of the function is interesting in theory since it allows to get proof of linear convergence. Following usual paradigms of numerical optimization, it would be possible to do better as soon as practical efficiency is concerned. An appropriate stopping rule still has to be found and studied; this is actually a general question and we are not aware of efficient techniques.

Note finally that the practical implementation of the regularization method of [75] does indeed something in-between: the simple scheme with one inner gradient iteration (second strategy) is used as a preprocessing phase before switching to making inner Newton iterations (first strategy). See more about this on numerical illustrations of the method in Sect. 20.4. A stopping test along the above lines would further enhance the numerical performance of the implementation.

#### 20.3.2.3 Illustration: Computing Lovász Theta Number

We finish this section with a numerical illustration (borrowed from [52]) of the performance of regularization methods on a classical combinatorial optimization problem.

Lovász [48] proved the celebrated “sandwich” theorem in graph theory: the stable number  $\alpha(G)$  of a graph  $G$  and the chromatic number  $\chi(\bar{G})$  of its complementary graph  $\bar{G}$  are separated

$$\alpha(G) \leq \vartheta(G) \leq \chi(\bar{G})$$

by the optimal value of an SDP problem

$$\vartheta(G) = \begin{cases} \max & \langle \mathbf{1}_{n \times n}, X \rangle \\ & X_{ij} = 0, \quad \text{when } (i, j) \text{ is an edge of } G \\ & \text{trace } X = 1, \quad X \succeq 0. \end{cases} \quad (20.44)$$

As expected, it can be shown that this SDP problem is a formulation of the SDP relaxation of the max-stable problem. The stable number  $\alpha(G)$  and the chromatic number  $\chi(\bar{G})$  are both NP-hard to compute and even hard to approximate, so that the tractable  $\vartheta(G)$  gives interesting information about  $G$ .

Some graphs from the DIMACS collection [39] are very challenging instances for computing  $\vartheta(G)$ . Those graphs have many edges and also many edges on

the complementary, so that makes them the most difficult for standard methods (as noticed in [22]). On the other hand, the structure of problem (20.44) is very favorable for regularization methods and in particular for the simple one of [52, Sect. 5.1] which is essentially (20.42)–(20.43). Observe indeed that the affine constraints (20.44) is given by “orthogonal” matrices (i.e.  $\langle A_j, A_i \rangle = 0$ ), such that the matrix  $AA^\top$  is diagonal. For illustration, the next table reports some of the bounds that were computed for the first time in [52]. For more examples, see [52, Sect. 5.4] and [75, Sect. 6.3].

Graph name	$n$	$m$	$\vartheta(G)$
brock400-1	400	59723	10.388
keller5	776	225990	31.000
brock800-1	800	207505	19.233
p-hat500-1	500	31569	58.036
p-hat1000-3	1000	371746	18.23

## 20.4 Applications to Polynomial Optimization

In this section, we illustrate the regularization methods for solving linear semidefinite optimization problems in the context of polynomial optimization. We collect numerical experiments showing that regularization algorithms can be considered as an alternative to standard methods for deciding whether a polynomial is non-negative (Sect. 20.4.2) and for globally minimizing a polynomial (Sect. 20.4.3). The point is thus the same as in [56] which reports extensive numerical results using regularization methods for solving various large-scale polynomial optimization problems.

We aim at giving here a methodology: our main focus is the generality of the approach and the reproducibility of experiments and results. We explain how to generate the test problems, we use public-domain implementations of the algorithms with default parameter tunings and with no attempt to adapt them to each particular problem instances (contrary to [56]). We do not carry out a comprehensive benchmarking with all methods and solvers; we just compare a widely used implementation of interior-point algorithm [69] with two recent implementations of regularization methods (the basic one of [52, Sect. 5.1], and the more sophisticated one of [75]).

### 20.4.1 Sum-of-Squares, SDP and Software

We briefly introduce in this section the notions and notation of polynomial optimization that we need. We refer to the recent surveys [44] and [43] and to the other chapters of this book for more on this topic.

Consider a multivariate polynomial of total degree  $2d$

$$v \in \mathbb{R}^N \longmapsto p(v) = \sum_{|\alpha| \leq 2d} p_\alpha v^\alpha. \quad (20.45)$$

We use here the multi-index notation  $v^\alpha = v_1^{\alpha_1} \cdots v_N^{\alpha_N}$  where  $\alpha \in \mathbb{N}^N$  runs over all nonnegative integer vectors of sum  $|\alpha| = \alpha_1 + \cdots + \alpha_N \leq 2d$ . We say that  $p(v)$  is a sum-of-squares (SOS) of polynomials if one could find polynomials  $q_k(v)$  such that

$$p(v) = \sum_k q_k^2(v). \quad (20.46)$$

It can be shown that finding such polynomials  $q_k(v)$  amounts to a semidefinite feasibility problem. More specifically, if  $\pi(v)$  denotes a vector of basis of polynomials of total degree less than or equal to  $d$ , finding an SOS decomposition (20.46) amounts to finding a so-called Gram matrix  $X \in \mathbb{R}^{n \times n}$  such that

$$p(v) = \pi(v)^\top X \pi(v) \quad \text{and} \quad X \in \mathcal{S}_n^+. \quad (20.47)$$

The set of SOS polynomials has thus a SDP representation of the form

$$Ax = b, \quad x \in \mathcal{K} \quad (20.48)$$

where  $\mathcal{K} = \mathcal{S}_n^+$ ,  $A \in \mathbb{R}^{m \times n^2}$  is a linear operator depending only on the choice of basis  $\pi(v)$ , and  $b \in \mathbb{R}^m$  is a vector depending on  $p(v)$ . For example if the vector of basis polynomials  $\pi(v) = [x^\alpha]_{|\alpha| \leq d}$  contains monomials  $x^\alpha$  indexed by  $\alpha \in \mathbb{N}^n$ , then identifying powers of  $v$  in relation (20.47) yields

$$p_\alpha = \langle A_\alpha, \pi(v) \pi(v)^\top \rangle, \quad \text{for all } \alpha$$

where matrix  $A_\alpha$  selects monomials  $x^\alpha$  in rank-one matrix  $\pi(v) \pi(v)^\top$ . More specifically, the entry in  $A_\alpha$  with row index  $\beta$  and column index  $\gamma$  is equal to one if  $\beta + \gamma = \alpha$  and zero otherwise. In problem (20.48), the row indexed by  $\alpha$  in matrix  $A$  collects entries of matrix  $A_\alpha$ , and the row indexed by  $\alpha$  in vector  $b$  is equal to  $p_\alpha$ . Note that

$$n = \binom{N+d}{N} \quad \text{and} \quad m = \binom{N+2d}{N},$$

so that the sizes of SDP problems grow quickly with the degree and the number of variables.

The important remark is that this type of constraints are favorable to regularization methods:  $AA^\top$  is always diagonal indeed. To see this, let  $\alpha, \beta$  denote the row and column indices in matrix  $AA^\top$ . By construction, the entry  $(\alpha, \beta)$  in  $AA^\top$  is equal to  $\langle A_\alpha, A_\beta \rangle$ : if  $\alpha = \beta$ , this is equal to the number of non-zero entries in matrix  $A_\alpha$ , otherwise, this is zero. Since it is important for numerical efficiency, we formalize the previous remark in a proposition.

**Proposition 20.3 (Orthogonality of constraints).** *Let  $A$  be the matrix in SOS semidefinite problem (20.48). Then  $AA^\top$  is diagonal with integer entries.*

Polynomial optimization problems that we consider in the next two sections are difficult to tackle directly but admit standard SOS relaxations involving constraints sets (20.48). In practice, an SOS relaxation approach boils down to solving linear semidefinite problems of the form

$$\begin{cases} \min & c^\top x \\ Ax = b, & x \in \mathcal{S}_n^+ \end{cases} \quad (20.49)$$

where and  $c \in \mathbb{R}^{n^2}$ ,  $b \in \mathbb{R}^m$ ,  $A \in \mathbb{R}^{m \times n^2}$ , and vector  $x$  collects entries of matrix  $X \in \mathbb{R}^{n \times n}$ . For solving problem (20.49), we use the three following the public-domain Matlab implementations:

1. SeDuMi1.3 implementing the primal-dual interior-point algorithm of [69] (available on [sedumi.ie.lehigh.edu](http://sedumi.ie.lehigh.edu))
2. MPRW a version of the basic regularization method of [52, Sect. 5.1] (available on [www.math.uni-klu.ac.at/or/Software/mprw2.m](http://www.math.uni-klu.ac.at/or/Software/mprw2.m))
3. SDPNAL0.1 the regularization method of [75] (available on [www.math.nus.edu.sg/~mattohkc/SDPNAL.html](http://www.math.nus.edu.sg/~mattohkc/SDPNAL.html))

Our goal here is just to show that the regularization methods are interesting in this context. We simply use default parameter tunings of the algorithms, with no attempt to adapt them to each particular problem instances contrary to in [56]. With  $K.s=n$ , the calling sequences of the three Matlab functions SeDuMi, MPRW and SDPNAL for solving (20.49) are thus as follows:

```

pars = [] ; pars.tol = 1e-9 ;
[x,y] = sedumi(A,b,c,K,pars) ;
X = reshape(x,K.s,K.s) ;

tol = 1e-9; C = reshape(c,K.s,K.s) ;
[X,y] = mprw(A,b,C,1e6,1,tol) ;

opts = [] ; opts.tol = 1e-9 ;
[blk,At,C,B] = read_sedumi(A,b,c,K) ;
[obj,X,y] = sdpnal(blk,At,C,B,opts) ;
X = X{1} ;

```

Experiments are carried out with Matlab 7.7 running on a Linux PC with Intel Xeon CPU W3520 2.67Ghz using 64 bit arithmetic and 8 GB RAM. Computation times are given in seconds, with two significant digits only (since our objective is not a comprehensive accurate benchmarking of codes).

Similarly to [56], we will see that, due to lower memory requirements, regularization methods can solve larger polynomial optimization problems than classical interior-point methods with the above setting.

A last note about tolerance. The tolerance parameters `tol` for the three solvers are set to  $10^{-9}$  for all the numerical experiments (except otherwise stated). Notice though that the meaning of the tolerance parameter is not the same for two types of algorithms. With regularization methods, the relative accuracy measured in terms of primal and residuals (remember (20.41)) is easily controlled. We stress that lower requirements on the relative accuracy could result in a significant saving of computational time, and this could be useful when solving approximately large-scale problems with MPRW and SDPNAL (see some examples in [56]). In contrast, we observe (as expected) that the iteration count of SeDuMi does not depend significantly on the expected accuracy, measured in terms of duality gap. Most of the computational time is spent to generate an approximately feasible primal-dual pair with relatively small duality gap, and only a few more iterations are required to refine the accuracy below  $10^{-9}$ .

### 20.4.2 Testing Positivity of Polynomials

We focus in this section on the very first problem of polynomial optimization: we would like to know whether a polynomial (20.45) is positive

$$p(v) \geq 0, \quad \text{for all } v \in \mathbb{R}^N. \quad (20.50)$$

In general this is a difficult problem for which no polynomial-time algorithm is known. It can be relaxed to the easier problem of testing if  $p$  could be expressed as an SOS (20.46). Whenever it holds, then obviously condition (20.50) is satisfied. The converse is not true in general if  $N \geq 2$  and  $d \geq 3$ , and there are explicit counter-examples; the simplest of them (the Motzkin polynomial) is studied below.

#### 20.4.2.1 Random Full-Rank Polynomial SOS Problems

We consider random polynomial SOS problems which are constructed so that there is a full-rank orthogonal Gram matrix  $X$  (an interior point) solving problem (20.47). We use GloptiPoly 3 (see [43]) to generate matrix  $A$  and vector  $b$  as follows:

```

N = 5; % number of variables
d = 3; % half degree
mpol('v',N,1); % variables
P = msdp(min((v'*v)^d)); % construct A matrix
[A,b,c,K] = msedumi(P); % retrieve A and K in SeDuMi
format
A = [c';-A]; % constant term and sign change
c = zeros(size(A,2),1); % no objective function
X = orth(randn(K.s)); % random Gram matrix
X = X'*diag(rand(K.s,1))*X;
b = A*X(:,1); % corresponding right handside vector

```

**Table 20.1** Comparative execution times for SOS problems

$N$	$n$	$m$	SeDuMi	MPRW	SDPNAL
5	56	462	0.29	0.03	0.05
6	84	924	0.92	0.05	0.07
7	120	1716	4.8	0.13	0.10
8	165	3003	25	0.35	0.16
9	220	5005	110	0.66	0.25
10	286	8008	410	1.3	0.43
11	364	12376	1500	3.0	0.73
12	455	18564	>3600	5.0	1.3

On Table 20.1 we report execution times (in seconds) required by SeDuMi, MPRW and SDPNAL to solve problem (20.48) for  $d = 3$  (degree six polynomials) and  $N = 5, \dots, 12$ . We also indicate the size  $n$  of matrix  $X$  and the number  $m$  of constraints (row dimension of matrix  $A$ ). We observe that SeDuMi is largely outperformed by MPRW and SDPNAL. We also observe that MPRW is about 4 times slower than SDPNAL, but this is not surprising as MPRW is a simple prototype (without comments and printing instructions it is about 50 lines of interpreted Matlab), whereas SDPNAL is a much more sophisticated package heavily relying on the efficient data handling and numerical linear algebra routines of the SDPT3 package. We also recall that SDPNAL makes several iterations of MPRW as preprocessing.

#### 20.4.2.2 Random Low-Rank Polynomial SOS Problems

We consider random polynomial SOS problems which are constructed so that there is a rank-one Gram matrix  $X$  solving problem (20.48). For such problems, it is unlikely that there is an interior point  $x$  solving problem (20.48), and indeed SeDuMi does not find a full-rank solution. We use the same code as above, replacing the instruction  $X = \text{orth}(\text{randn}(K.s)); X = X'*\text{diag}(\text{rand}(K.s, 1))*X;$  with the instructions

```
X = orth(randn(K.s, 1));
X = X*X';
```

We report execution times (in seconds) in Table 20.2. In comparison with the problems with interior points of Table 20.1, we observe that all the solvers experience convergence issues. However, there is still a considerable improvement in terms of efficiency brought by regularization methods, though Slater's qualification constraint cannot be invoked to guarantee convergence.

#### 20.4.2.3 Motzkin's Polynomial

We study a well-known bivariate ( $N = 2$ ) polynomial of sixth degree ( $d = 3$ ) which is non-negative but cannot be written as a polynomial SOS, namely Motzkin's polynomial

**Table 20.2** Comparative execution times for low-rank SOS problems

<i>N</i>	<i>n</i>	<i>m</i>	SeDuMi	MPRW	SDPNAL
5	56	462	0.83	0.20	0.21
6	84	924	0.85	0.32	0.28
7	120	1716	16	2.1	0.51
8	165	3003	61	4.8	0.98
9	220	5005	330	12	1.2
10	286	8008	1300	24	2.5
11	364	12376	>3600	50	3.5
12	455	18564	>3600	110	6.6

$$p_0(v) = 1 + v_1^2 v_2^2 (v_1^2 + v_2^2 - 3)$$

see [44] or [43]. This polynomial achieves its minimum zero at the four points  $v_1 = \pm 1, v_2 = \pm 1$ . In a basis of monomials of degree up to 3 there is no Gram matrix  $X$  solving problem (20.48). However, it was observed in [33] and later on shown theoretically in [42] that the perturbed polynomial

$$p_0(v) + \varepsilon p_1(v)$$

can be represented as a polynomial SOS (with full-rank Gram matrix) provided the degree of the perturbation polynomial  $p_1(v)$  is high enough, inversely proportional to scalar  $\varepsilon > 0$ . In some sense, this can be interpreted as a regularization procedure as in [34]. Practically speaking, since semidefinite programming solvers use inexact operations (floating point arithmetic), it is not necessary to perturb explicitly the data. It is enough to choose a basis  $\pi(v)$  of sufficiently high degree  $d > 3$  in relation (20.47), and higher-order perturbations are automatically introduced by the algorithm.

We use the following GoptiPoly3 instructions to generate data  $A, b$  for increasing values of  $d$ :

```

d = 8; % half degree
mpol v1 v2
p = 1+v1^2*v2^2*(v1^2+v2^2-3);
P = msdp(min(p),d);
[A,b,c,K,b0] = msedumi(P);
A = [c';-A];
b = [-b0;-b];
c = zeros(size(A,2),1);

```

For this problem, we set `tol=1e-6` for the three solvers. When  $d = 3, 4, 5, 6$ , SeDuMi takes less than 0.1 s to detect that problem (20.48) is infeasible, and it provides a Farkas dual certificate vector  $y \in -\mathcal{K}$  such that  $b^\top y = 1$ . When  $d = 7, 8, 9, 10$ , SeDuMi takes less than 0.5 s to return a vector  $x$  such that the primal residual  $\|Ax - b\|_2/\|b\|_2$  is less than  $10^{-9}$  and the dual objective function  $b^\top y$  is less than  $10^{-9}$  in absolute value.

**Table 20.3** Behavior of MPRW (left) and SDPNAL (right) for Motzkin's polynomial

$d$	time	$\ Ax - b\ _2 / \ b\ _2$	$b^\top y$
3	—	—	—
4	—	—	—
5	—	—	—
6	15	$6.20 \cdot 10^{-6}$	$1.12 \cdot 10^{-6}$
7	25	$6.03 \cdot 10^{-7}$	$6.81 \cdot 10^{-7}$
8	26	$5.80 \cdot 10^{-6}$	$-4.08 \cdot 10^{-7}$
9	34	$1.01 \cdot 10^{-6}$	$-1.45 \cdot 10^{-7}$
10	75	$5.42 \cdot 10^{-7}$	$-1.58 \cdot 10^{-7}$
3	5.1	$4.28 \cdot 10^{-3}$	33.4
4	9.2	$1.56 \cdot 10^{-4}$	0.832
5	3.5	$4.59 \cdot 10^{-6}$	$4.37 \cdot 10^{-5}$
6	4.6	$6.33 \cdot 10^{-6}$	$1.05 \cdot 10^{-6}$
7	5.7	$8.95 \cdot 10^{-6}$	$3.86 \cdot 10^{-7}$
8	5.9	$2.79 \cdot 10^{-6}$	$-3.46 \cdot 10^{-7}$
9	7.9	$2.54 \cdot 10^{-6}$	$-3.25 \cdot 10^{-7}$
10	8.8	$1.88 \cdot 10^{-6}$	$-1.34 \cdot 10^{-7}$

The behavior of SDPNAL and MPRW is more erratic, and convergence issues occur, as shown by the execution times (in seconds) of Table 20.3. For  $d = 3, 4, 5$ , MPRW stops after  $10^6$  iterations, as there is no mechanism to detect infeasibility in this prototype software.

#### 20.4.2.4 Regularization vs Projection

Though it solves linear semidefinite problems, using regularization techniques somehow generalizes and enhances the idea [34] to using projection methods directly for SOS feasibility problems. With this approach indeed, a question is to find a good point to project; taking systematically the zero matrix gives interesting results but could be greatly enhanced. Regularization methods provide a numerical solution to this: doing a sequence of truncated projections allows to keep the projection idea while getting rid of the question of the initial point to project. The behaviour of SDPNAL is interesting with this respect: it does first a preprocessing of several alternating direction iterations to get a meaningful point, then follows by projection-like iterations. In practice, we observe usually a very few iterations, and often one. For example, to decide whether the (admittedly trivial) polynomial  $p(v) = \sum_{i=1}^{10} v_i^{10}$  is SOS, the SDP problem dimensions are  $n = 3003$  and  $m = 184756$ , and after 90 s and only one projection-like iteration, SDPNAL provides a vector  $x$  satisfying  $\|Ax - b\|_2 / \|b\|_2 \approx 1.4 \cdot 10^{-10}$ .

### 20.4.3 Unconstrained Polynomial Minimization

In this section we study global minimization problems

$$p^* = \min_{v \in \mathbb{R}^N} p(v) \quad (20.51)$$

where  $p(v)$  is a given polynomial. For this problem, a semidefinite relaxation readily follows from the observation that

$$\begin{aligned} p^* &= \max_p \underline{p} \\ \text{s.t. } & p(v) - \underline{p} \geq 0, \quad \forall v \in \mathbb{R}^N \end{aligned}$$

and by relaxing the above non-negativity constraint by the semidefinite programming constraint that polynomial  $p(v) - \underline{p}$  is SOS, see [44] and [43].

#### 20.4.3.1 Random Polynomial Minimization Problems

We generate well-behaved instances of unconstrained polynomial minimization problems (20.51) with

$$p(v) = p_0(v) + \sum_{i=1}^N v_i^{2d}$$

where  $p_0(v)$  is a random polynomial of total degree strictly less than  $2d$ . The leading term  $\sum_{i=1}^N v_i^{2d}$  ensures coercivity of  $p(v)$  and hence existence of a global minimum in (20.51). We use the following GloptiPoly 3 script to generate our examples:

```
N = 10;
mpol('v',N,1);
b = mmon(v,0,2*d-1); % degree up to 2d-1
p0 = randn(1,length(b)); p0 = p0/norm(p0);
p = p0*b + sum(mmon(v,d).^2);
P = msdp(min(p));
[A,b,c,K] = msedumi(P);
```

In Table 20.4 we report comparative execution times (in seconds) for  $d = 2$  and various values of  $N$ , for solving the semidefinite relaxation. It turns out that for these generic problems, we observe that global optimality is always certified with a rank-one moment matrix [33]. Both MPRW and SDPNAL largely outperform SeDuMi on these examples.

**Table 20.4** Comparative execution times for semidefinite relaxations of random polynomial minimization problems

$N$	$n$	$m$	SeDuMi	MPRW	SDPNAL
5	21	126	0.09	0.05	0.18
6	28	209	0.11	0.07	0.18
7	36	329	0.24	0.12	0.20
8	45	494	0.36	0.19	0.22
9	55	714	0.77	0.28	0.26
10	66	1000	1.9	0.45	0.29
11	78	1364	5.0	0.78	0.36
12	91	1819	11	1.1	0.41
13	105	2379	20	1.6	0.47
14	120	3059	42	2.3	0.65
15	136	3875	74	3.0	0.68

**Table 20.5** Comparative execution times for semidefinite relaxations of a larger polynomial minimization problem

$N$	$n$	$m$	SeDuMi	MPRW	SDPNAL
5	56	461	0.50	0.54	0.60
6	84	923	2.5	1.2	1.2
7	120	1715	14	5.0	2.6
8	165	3002	92	19	6.7
9	220	5004	410	65	22
10	286	8007	1800	200	71
11	364	12375	7162	490	150
12	455	18563	> 7200	1500	530
13	560	27131	> 7200	3500	2300
14	680	38760	> 7200	> 7200	9900

### 20.4.3.2 A Structured Example

Consider the problem studied in [56, Example 3.5], that is (20.51) with

$$p(v) = \sum_{i=1}^N \left( 1 - \sum_{j=1}^i (v_j + v_j^2) \right)^2 + \left( 1 - \sum_{j=1}^N (v_j + v_j^3) \right)^2.$$

We solve the semidefinite relaxation for increasing values of  $N$ . We collect comparative execution times on Table 20.5 for this example. For example, when  $N = 10$ , SDPNAL resp. MPRW returns a point  $x$  such that  $\|Ax - b\|/\|b\|$  is equal to  $1.4 \cdot 10^{-9}$  resp.  $2.6 \cdot 10^{-10}$  and the minimum eigenvalue of  $X$  is equal to zero to machine precision.

We observe again a considerable improvement in terms of performance brought by regularization methods in comparison with a classical interior-point method. For larger instances, most of the computation time of SeDuMi is spent for memory swapping when constructing and handling large matrices. We refer to the recent extensive numerical work of [56] for various structured problems.

**Acknowledgement** The work of the first author was partly supported by research project 103/10/0628 of the Grant Agency of the Czech Republic and research programme MSM6840770038 of the Ministry of Education of the Czech Republic.

## References

1. Al-Homidan, S.: Solving Hankel matrix approximation problem using semidefinite programming. *Journal of Computational and Applied Mathematics* **202**(2), 304–314 (2007)
2. Alizadeh, F., Haeberly, J.-P., Overton, M.: Complementarity and nondegeneracy in semidefinite programming. *Math. Program.* **77**(2), 111–128 (1997)
3. Anjos, M.F., Higham, N.J., Takouda, P.L., Wolkowicz, H.: A semidefinite programming approach for the nearest correlation matrix problem. Technical report, Combinatorics and Optimization, University of Waterloo, September 2003.
4. Arrow, K., Hurwicz, L., Uzawa, H.: *Studies in Linear and Nonlinear Programming*. Stanford University Press (1959)
5. Bolte, J., Daniilidis, A., Lewis, A.: Tame functions are semismooth. *Math. Program.* **117**(1), 5–19 (2008)
6. Bertsekas, D.P.: *Nonlinear Programming*. Athena Scientific (1995)
7. Boyd, S., El Ghaoui, L., Feron, E., Balakrishnan, V.: *Linear Matrix Inequalities in System and Control Theory*. Studies in Applied Mathematics. SIAM (1994)
8. Bonnans, J.F., Gilbert, J.Ch., Lemaréchal, C., Sagastizábal, C.: *Numerical Optimization*. Springer Verlag (2003)
9. Borsdorf, R., Higham, N.: A preconditionned Newton algorithm for the nearest correlation matrix. *IMA J. Numer. Anal.* **30**(1), 94–107 (2010)
10. Bellman, R., Kalaba, R., Lockett, J.: *Numerical Inversion of the Laplace Transform*. Elsevier (1966)
11. Brigo, D., Mercurio, F.: *Interest rate models: theory and practice*. Springer-Verlag (2006)
12. Burer, S., Monteiro, R., Zhang, Y.: A computational study of a gradient-based log-barrier algorithm for a class of large-scale sdps. *Mathematical Programming Series B*, 359–379, 2001.
13. Bonnans, J.F., Shapiro, A.: *Perturbation Analysis of Optimization Problems*. Springer Verlag (2000)
14. Boyd, S., Vandenberghe, L.: *Convex optimization*. Cambridge University Press (2004)
15. Burer, S., Vandenbussche, D.: Solving lift-and-project relaxations of binary integer programs. *SIAM Journal on Optimization* **16**, 726–750 (2006)
16. Boyd, S., Xiao, L.: Least-squares covariance matrix adjustment. *SIAM Journal on Matrix Analysis and Applications* **27**(2) (2005)
17. Correa, R., Lemaréchal, C.: Convergence of some algorithms for convex minimization. *Mathematical Programming* **62**(2), 261–275 (1993)
18. Clarke, F.H.: *Optimization and Nonsmooth Analysis*. Wiley (1983); reprinted by SIAM, (1983)
19. Dembo, R., Eisenstat, S., Steihaug, T.: Inexact Newton methods. *SIAM Journal on Numerical Analysis* **19**(2) (1982)
20. Deutsch, F.: *Best Approximation in Inner Product Spaces*. Springer, New York (2001)
21. Douglas, J., Rachford, H.H.: On the numerical solution of heat conduction problems in two and three space variables. *Trans. Amer. Math. Soc.* **82**, 421–439 (1956)
22. Dukanovic, I., Rendl, F.: Semidefinite programming relaxations for graph coloring and maximal clique problems. *Mathematical Programming* **109**, 345–365 (2007)
23. Dumitrescu, B.: *Positive Trigonometric Polynomials and Signal Processing Applications*. Springer Verlag (2007)
24. Dykstra, R.L.: An algorithm for restricted least-square regression. *Journal of the American Statistical Association* **78**, 837–842, (1983)

25. Gilbert, J.Ch., Lemaréchal, C.: Some numerical experiments with variable-storage quasi-Newton algorithms. *Mathematical Programming* **45**, 407–435 (1989)
26. Gabay, D., Mercier, B.: A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. *Computers and Mathematics with Applications* **2** (1976)
27. Gao, Y., Sun, D.: Calibrating least squares semidefinite programming with equality and inequality constraints. *SIAM J. Matrix Anal. Appl.* **31**(3), 1432–1457 (2009)
28. Gao, Y., Sun, D.: A majorized penalty approach for calibrating rank constrained correlation matrix problems. Technical report, Univ. of Singapore (2003)
29. Goemans, M., Williamson, D.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM* **6**, 1115–1145 (1995)
30. Henrion, D., Garulli, A. (eds.): Positive polynomials in control. LNCIS, Springer Verlag (2005)
31. Higham, N.: Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and its Applications* **103**, 103–118 (1988)
32. Higham, N.: Computing a nearest symmetric correlation matrix — a problem from finance. *IMA Journal of Numerical Analysis* **22**(3), 329–343 (2002)
33. Henrion, D., Lasserre, J.-B.: Detecting global optimality and extracting solutions in gloptipoly. In: Henrion, D., Garulli, A., (eds.) Positive polynomials in control, LNCIS. Springer (2005)
34. Henrion, D., Malick, J.: Projection methods for conic feasibility problems; application to sum-of-squares decompositions. *Optimization Methods and Software* **26**(1), 23–46 (2011)
35. Hiriart-Urruty, J.-B. Lemaréchal, C.: Convex Analysis and Minimization Algorithms. Springer Verlag, Heidelberg (1993) Two volumes.
36. Hiriart-Urruty, J.-B., Stoeriot, J.J., Nguyen, H.V.: Generalized hessian matrix and second-order optimality conditions for problems with  $C^{1,1}$  data. *Applied Mathematics and Optimization* **11**, 43–56 (1984)
37. He, B., Xu, M., Yuan, X.: Solving large-scale least-squares covariance matrix problems by alternating direction methods. Technical report (2009)
38. Jarre, F., Rendl, F.: An augmented primal-dual method for linear conic problems. *SIAM Journal on Optimization* (2008)
39. Johnson, D.J., Trick, M.A. (eds.): Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, Workshop, 11-13 October 1993. American Mathematical Society, Boston, MA, USA (1996)
40. Kočvara, M., Stingl, M.: PENNON: a code for convex nonlinear and semidefinite programming. *Optimisation Methods and Sofware* **18**(3), 317–333 (2003)
41. Kočvara, M., Stingl, M.: On the solution of large-scale sdp problems by the modified barrier method using iterative solvers. *Math. Program.* **109**(2), 413–444 (2007)
42. Lasserre, J.-B.: A sum of squares approximation of nonnegative polynomials. *SIAM J. Optim.* **16**, 751–765 (2006)
43. Lasserre, J.-B.: Moments, Positive Polynomials and Their Applications. Imperial College Press (2009)
44. Laurent, M.: Sums of squares, moment matrices and optimization over polynomials. In: M. Putinar and S. Sullivant, editors, Emerging Applications of Algebraic Geometry, IMA Volumes in Mathematics and its Applications. Springer (2009)
45. Lin, F., Jovanović, M.R.: Least-squares approximation of structured covariances. *IEEE Trans. Automat. Control* **54**(7), 1643–1648 (2009)
46. Lewis, A., Luke, D., Malick, J.: Local linear convergence for alternating and averaged nonconvex projections. *Foundations of Computational Mathematics* **9**, 485–513 (2009)
47. Lewis, A., Malick, J.: Alternating projections on manifolds. *Mathematics of Operations Research* **33**(1), 216–234 (2008)
48. Lovász, L.: On the Shannon capacity of a graph. *IEEE Transactions on Information Theory* **25**, 1–7 (1979)
49. Li, Q., Qi, H.: A sequential semismooth Newton method method for the nearest low-rank correlation matrix problem. Technical report (2003)

50. Malick, J.: A dual approach to semidefinite least-squares problems. *SIAM Journal on Matrix Analysis and Applications* **26**(1), 272–284 (2004)
51. B. Martinet. Régularisation d'inéquations variationnelles par approximations successives. *Revue Française d'Informatique et Recherche Opérationnelle* **R-3**, 154–179 (1970)
52. Malick, J., Povh, J., Rendl, F., Wiegele, A.: Regularization methods for semidefinite programming. *SIAM Journal on Optimization* **20**(1), 336–356 (2009)
53. Malick, J., Sendov, H.: Clarke generalized Jacobian of the projection onto the cone of positive semidefinite matrices. *Set-Valued Analysis* **14**(3), 273–293 (2006)
54. Micchelli, C.A., Utreras, F.I.: Smoothing and interpolation in a convex subset of an Hilbert space. *SIAM J. Sci. Stat. Comput.* **9**, 728–746 (1988)
55. Nesterov, Y.: Smooth minimization of non-smooth functions. *Math. Program.* **103**(1), 127–152 (2005)
56. Nie, J.: Regularization methods for sum of squares relaxations in large scale polynomial optimization. Technical report, ArXiv (2009)
57. Caprara, A., Oswald, M., Reinelt, G., Schwarz, R., Traversi, E.: Optimal linear arrangements using betweenness variables. *Mathematical Programming (Series C)* **3**(3), 261–280, (2011)
58. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Verlag, New York (2006)
59. Nouralishahi, M., Wu, C., Vandenberghe, L.: Model calibration for optical lithography via semidefinite programming. *Optimization and Engineering* **9**, 19–35 (2008)
60. Povh, J., Rendl, F., Wiegele, A.: A boundary point method to solve semidefinite programs. *Computing* **78**, 277–286 (2006)
61. Polyak, B.T., Tretjakov, N.V.: On an iterative method for linear programming and its economical interpretations. *Ekonom. i Mat. Methody* **8**, 740–751 (1972)
62. Qi, L.Q., Sun, J.: A nonsmooth version of Newton's method. *Mathematical Programming* **58**(3), 353–367 (1993)
63. Qi, H., Sun, D.: Quadratic convergence and numerical experiments of Newton's method for computing the nearest correlation matrix. *SIAM Journal on Matrix Analysis and Applications* **28**, 360–385 (2006)
64. Qi, H., Sun, D.: An augmented Lagrangian dual approach for the h-weighted nearest correlation matrix problem. *IMA Journal of Numerical Analysis* (2003)
65. Rockafellar, R.T.: Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research* **1**, 97–116 (1976)
66. Rockafellar, R.T.: Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization* **14**, 877–898 (1976)
67. Schwertman, N., Allen, D.: Smoothing an indefinite variance-covariance matrix. *Journal of Statistical Computation and Simulation* **9**, 183–194 (1979)
68. Sun, D., Sun, J.: Semismooth matrix valued functions. *Mathematics of Operations Research* **57** (2002)
69. Sturm, J.F.: Using Sedumi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Softwares* **11-12**, 625–653 (1999)
70. Saigal, R., Vandenberghe, L., Wolkowicz, H.: Handbook of Semidefinite-Programming. Kluwer (2000)
71. Sun, J., Zhang, S.: A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs. *European Journal of Operational Research* **207**(3), 1210–1220 (2003)
72. Toh, K.: An inexact primal-dual path following algorithm for convex quadratic SDP. *Math. Program.* **112**(1), 221–254 (2008)
73. Tutuncu, R.H., Toh, K.C., Todd, M.J.: Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming B* **95**, 189–217 (2003)
74. Tutuncu, R.H., Toh, K.C., Todd, M.J.: Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems. *Pacific Journal on Optimization* (2006)
75. Zhao, X., Sun, D., Toh, K.: A Newton-CG augmented lagrangian method for semidefinite programming. *SIAM J. Optim* **20**(4) (2003)

# Chapter 21

## SDP Relaxations for Non-Commutative Polynomial Optimization

Miguel Navascués, Stefano Pironio, and Antonio Acín

### 21.1 Introduction

Polynomial optimization problems involving operator (i.e., non-commuting) variables arise naturally in many areas of quantum physics. Let us give some examples that motivate the present chapter.

#### 21.1.1 Examples of Polynomial Optimization Problems

##### 21.1.1.1 Quantum Correlations Between Separated Systems

Let  $M_1, \dots, M_N$  be a collection of finite disjoint sets. Each  $M_k$  represents a measurement that can be performed on a given system and the elements  $i \in M_k$  are the possible outcomes of the  $k$ -th measurement. We suppose that the system is composed of two non-interacting subsystems and that measurements

---

M. Navascués (✉)

Department of Mathematics, University of Bristol, University Walk, BS8 1TW, Bristol, United Kingdom

e-mail: [m.navascues@bristol.ac.uk](mailto:m.navascues@bristol.ac.uk)

S. Pironio

Laboratoire d'Information Quantique, C.P. 225, Av. F. D. Roosevelt 50, B-1050 Bruxelles, Belgium

e-mail: [stefano.pironio@ulb.ac.be](mailto:stefano.pironio@ulb.ac.be)

A. Acín

ICFO-Institut de Ciències Fotòniques, Mediterranean Technology Park, 08860 Castelldefels, Spain

ICREA-Institució Catalana de Recerca i Estudis Avançats, Passeig Lluís Companys 23, 08010 Barcelona, Spain

e-mail: [antonio.acin@icfo.es](mailto:antonio.acin@icfo.es)

$M_1, \dots, M_n$  are performed on the first system and measurements  $M_{n+1}, \dots, M_N$  on the second. We write  $A = M_1 \cup \dots \cup M_n$  and  $B = M_{n+1} \cup \dots \cup M_N$ . The correlations between the two systems are characterized by the joint probabilities  $P(i,j)$  to obtain outcomes  $i \in A$  and  $j \in B$ . In quantum theory, these probabilities are given by  $P(i,j) = \langle \phi, E_i E_j \phi \rangle$ , where  $\phi$  describes the joint state of the two systems and the operators  $E_i$  describe the measurements performed on  $\phi$ . The operators  $\{E_i : i \in M_k\}$  associated to the measurement  $M_k$  form an orthogonal resolution of the identity, and operators corresponding to different subsystems commute, i.e.,  $[E_i, E_j] = 0$  when  $i \in A$  and  $j \in B$ .

Physical quantities of interest are often linear in the joint probabilities  $P(i,j)$ . Examples are the amount of violation of a Bell inequality [3] or the probability that two separated players can successfully perform a distributed computation using quantum resources [8]. One is then interested in finding the quantum state  $\phi$  and operators  $E_i$  that maximize this linear quantity, that is, in solving the problem

$$\max_{E, \phi} \quad \left\langle \phi, \sum_{ij} c_{ij} E_i E_j \phi \right\rangle$$

subject to  $\|\phi\| = 1$ ,

$$\begin{aligned} E_i E_j &= \delta_{ij} E_i && \text{for all } i, j \in M_k \text{ and for all } M_k, \\ \sum_{i \in M_k} E_i &= 1 && \text{for all } M_k, \\ [E_i, E_j] &= 0 && \text{for all } i \in A \text{ and for all } j \in B. \end{aligned} \tag{21.1}$$

### 21.1.1.2 Quantum Chemistry

Quantum chemistry is concerned with the electronic structure and properties of atoms and molecules [57]. Consider for instance an atom comprised of  $N$  electrons that can occupy  $M$  orbitals, each orbital being associated with annihilation and creation operators  $a_i$  and  $a_i^*$ ,  $i = 1, \dots, M$ . Since electrons interact pairwise, the hamiltonian is characterized by two-body interaction terms  $h_{ijkl}$  and the ground state energy of the system can be computed as

$$\min_{a, a^*, \phi} \quad \left\langle \phi, \sum_{ijkl} h_{ijkl} a_i^* a_j^* a_k a_l \phi \right\rangle$$

subject to  $\|\phi\| = 1$ ,

$$\{a_i, a_j\} = 0 \quad \text{for all } i, j,$$

$$\{a_i^*, a_j^*\} = 0 \quad \text{for all } i, j,$$

$$\{a_i^*, a_j\} = \delta_{ij} \quad \text{for all } i, j,$$

$$\left( \sum_i a_i^* a_i - N \right) \phi = 0. \tag{21.2}$$

The first three constraints on the operators  $a_i, a_i^*$  represent the usual anticommutation fermionic relations, while the last constraint fixes the number of electrons to  $N$ . Variants of the above problem can be used to compute the energy of excited levels or orbital occupation numbers.

### 21.1.1.3 Interacting Spin Systems

Consider  $N$  interacting spin 1/2 particles on a chain in the presence of an external magnetic field. The hamiltonian describing the system is given by

$$H(S) = \sum_{jk} J_{jk} (S_j^x S_k^x + S_j^y S_k^y + S_j^z S_k^z) + \sum_j h_j S_j^z,$$

where  $J_{jk}, h_j$  are real constants and  $S_j^a$  ( $a = x, y, z$ ) are the Pauli operators for particle  $j$  defined by  $S_j^a S_j^b = \delta_{ab} + i \sum_c \epsilon_{abc} S_j^c$ , where  $\epsilon_{abc}$  is the Levi–Civita symbol. The minimal value of the correlation function  $\frac{1}{N} \langle \sum_j S_j^z S_{j+1}^z \rangle$  for an average energy  $E$  can be computed as

$$\min_{S, \phi} \quad \frac{1}{N} \left\langle \phi, \sum_j S_j^z S_{j+1}^z \phi \right\rangle$$

subject to  $\|\phi\| = 1$ ,

$$\begin{aligned} S_j^a S_j^b &= \delta_{ab} + i \sum_c \epsilon_{abc} S_j^c && \text{for all } i \text{ and } a, b = x, y, z, \\ [S_j^a, S_k^b] &= 0 && \text{for all } j \neq k. \end{aligned} \tag{21.3}$$

$$\langle \phi, H(S) \phi \rangle = E.$$

### 21.1.2 Generic Problem

The above problems have the generic form:

$$\min_{X, \phi} \quad \langle \phi, p(X) \phi \rangle$$

subject to  $\|\phi\| = 1$ ,

$$\begin{aligned} q_i(X) &\geq 0 & i &= 1, \dots, m_q, \\ r_i(X)\phi &= 0 & i &= 1, \dots, m_r, \\ \langle \phi, s_i(X)\phi \rangle &\geq 0 & i &= 1, \dots, m_s, \end{aligned} \tag{21.4}$$

where  $X = (X_1, \dots, X_n)$  is a set of bounded operators on a Hilbert space  $H$ ,  $\phi$  is a normalized vector in  $H$ ,  $p(X), q_i(X), r_i(X), s_i(X)$  are polynomials in the operator variables  $X$  (e.g.,  $p(X) = X_1 X_2 + 4X_3^2$ ), and where the condition  $q_i(X) \geq 0$  means that the operator  $q_i(X)$  is positive semidefinite. The optimization is performed over all operators  $X$  and vectors  $\phi$  (defined in Hilbert spaces  $H$  of arbitrary dimension) compatible with the constraints.

Such problems can be seen as a generalization of usual polynomial optimization problems in scalar (commuting) variables to the case of operator (non-commuting) variables. In recent years, hierarchies of semidefinite programming (SDP) relaxations for polynomial optimization problems, e.g. as introduced by Lasserre [31] and Parrilo [47], have generated much interest [32]. The aim of this work is to show how similar SDP relaxations can be introduced to deal with non-commutative (NC) problems of the form (21.4).

### 21.1.3 The Key Connection

To explain the connection with the sequence of SDP relaxations introduced by Lasserre, consider the following instance of (21.4),

$$\begin{aligned} \min_{X, \phi} \quad & \langle \phi, p(X)\phi \rangle \\ \text{subject to} \quad & \|\phi\| = 1, \\ & q_i(X) \geq 0 \quad i = 1, \dots, m, \end{aligned} \tag{21.5}$$

where  $X = (X_1, \dots, X_n)$  are assumed to be symmetric, bounded operators on a real Hilbert space. If we impose the additional condition that the operators  $X$  commute, i.e.,  $X_i X_j = X_j X_i$ , then the problem (21.5) reduces to the scalar problem

$$\begin{aligned} \min_x \quad & p(x) \\ \text{subject to} \quad & q_i(x) \geq 0 \quad i = 1, \dots, m, \end{aligned} \tag{21.6}$$

in the variables  $x = (x_1, \dots, x_n) \in \mathbb{R}^n$  [49]. Indeed, if the operators  $X$  in (21.5) commute, they generate an abelian algebra that is unitarily equivalent, by the spectral theorem, to the algebra of diagonal operators on the direct integral Hilbert space  $\int^{\oplus} H_x d\mu(x)$  [61]. The problem (21.5) is then equivalent to  $\min_{\mu} \int_K p(x) d\mu(x)$  subject to  $\int_K d\mu(x) = 1$  where  $K = \{x \in \mathbb{R}^n : q_i(x) \geq 0, i = 1, \dots, m\}$  [49], which is in turn equivalent to (21.6), as shown by Lasserre [31].

The basic idea underlying Lasserre's method for solving (21.6) is as follows. Let  $w(x)$  denote a monomial, e.g.,  $w(x) = x_1^2 x_2$ . The polynomial  $p(x)$  can then be decomposed in the basis of monomials as  $p(x) = \sum_w p_w w(x)$ . Treat now every  $w(x)$  as an independent variable  $y_w$ , and instead of minimizing  $p(x)$ , minimize the linear

combination  $\sum_w p_w y_w$ . Of course, if each  $y_w$  is to be interpreted as a monomial  $w(x)$ , the  $y_w$ 's should not be independent but satisfy non-convex polynomial constraints. For instance if  $v, w$  are monomials and  $vw$  is their product (which is also a monomial), then we should have  $y_{vw} = y_v y_w$ . Such non-convex constraints are relaxed by requiring instead the positivity of a “moment” matrix  $M(y)$ , whose entries are indexed in the basis of monomials and are given by  $M(y)(v, w) = y_{vw}$ . Indeed, if  $y_w$  is of the desired form

$$y_w = w(x) \quad (21.7)$$

for some  $x$ , then  $M(y) \geq 0$  since for any vector  $z$ ,

$$\begin{aligned} z^* M(y) z &= \sum_{v,w} z_v M(y)(v, w) z_w = \sum_{v,w} z_v y_{vw} z_w \\ &= \sum_v z_v v(x) \sum_w z_w w(x) = z(x)^2 \geq 0. \end{aligned} \quad (21.8)$$

Similarly, the constraints  $q_i(x) \geq 0$  are relaxed by imposing the positivity of a set of “localizing matrices”  $M(q_i y)$  defined by  $M(q_i y)(v, w) = \sum_u q_u y_{vuw}$ . The problem (21.6) can then be replaced by

$$\begin{aligned} \min_y \quad & \sum_w p_w y_w \\ \text{subject to} \quad & M(y) \geq 0, \\ & M(q_i y) \geq 0 \quad i = 1, \dots, m, \end{aligned} \quad (21.9)$$

which is a semidefinite program. The matrices  $M(y)$  and  $M(q_i y)$  are infinite, but we can truncate them at any finite size  $k$  and require the positivity of the truncated matrices. This defines, for increasing  $k$ , a hierarchy of SDP relaxations, which can be shown to converge to (21.6) under suitable conditions. The duals of these SDP relaxations are related to the theory of sums of squares (SOS) and their convergence can be established using a SOS representation theorem of Putinar for positive polynomials [51].

The above approach can be adapted to the problem (21.5) as follows. As above, let  $w(X)$  denote a monomial (e.g.,  $w(X) = X_1^2 X_2$ ) and expand the polynomial  $p(X)$  as  $p(X) = \sum_w p_w w(X)$  in the basis of monomials. Note that the basis of monomials is larger than in the previous case, since the operators  $X_i$  do not commute in general. For instance, the set of monomials of degree smaller than 2 in the two variables  $X_1, X_2$  is  $\{1, X_1, X_2, X_1^2, X_1 X_2, X_2 X_1, X_2^2\}$  instead of  $\{1, x_1, x_2, x_1^2, x_1 x_2, x_2^2\}$ , in the previous, commutative case.

One then treats the quantities  $\langle \phi, w(X) \phi \rangle$  as independent variables  $y_w$  and replaces the minimization of  $\langle \phi, p(X) \phi \rangle = \sum_w p_w \langle \phi, w(X) \phi \rangle$  by the minimization of the linear combination  $\sum_w p_w y_w$ . To constrain further the  $y_w$ 's, we define by analogy with the previous case, a “moment” matrix  $M(y)$  with entries  $M(y)(v, w) = y_{v^* w}$ , where  $v^*(X)$  is the adjoint of  $v(X)$  and is itself a monomial (e.g.,  $v^*(X) = X_2 X_1^2$  if  $v(X) = X_1^2 X_2$ ). If the  $y_w$ 's are of the desired form

$$y_w = \langle \phi, w(X) \phi \rangle \quad (21.10)$$

for some operators  $X$  and a vector  $\phi$ , then the moment matrix  $M(y)$  should be positive semidefinite since for any vector  $z$ ,

$$\begin{aligned} z^* M(y) z &= \sum_{v,w} z_v^* M(y)(v,w) z_w = \sum_{v,w} z_v^* y_{v^*w} z_w \\ &= \left\langle \phi, \sum_v z_v^* v^*(X) \sum_w z_w w(X) \phi \right\rangle = \langle \phi, z(X)^* z(X) \phi \rangle \geq 0. \end{aligned} \quad (21.11)$$

Similarly, one can show that the “localizing” matrices  $M(q_i y)$  defined through  $M(q_i y)(v,w) = \sum_u q_u y_{v^*uw}$  should also be positive semidefinite. As before, the problem (21.5) can then be replaced by a SDP of the form (21.9), which generates a hierarchy of SDP relaxations through finite truncations of increasing size [49]. The only difference between the SDP relaxations for the problem (21.5) and (21.6) lies in the monomial basis used to expand the polynomials  $p, q_i$  and to define the moment and localizing matrices: it distinguishes  $X_i X_j$  and  $X_j X_i$  in the former case, while  $x_i x_j = x_j x_i$  in the latter case. In this sense, the SDP relaxations associated to the problem (21.5) represent the proper non-commutative analogue of the SDP relaxations introduced by Lasserre for scalar problems.

### 21.1.4 SDP Relaxations for NC Polynomial Optimization

This hierarchy of SDP relaxations is presented in full detail in Sect. 21.2. We describe it for the more general problem (21.4) and the case of non-hermitian operators, to which the method that we briefly sketched easily extends. The optimal solutions of these relaxations form a monotonically increasing sequence of lower bounds on the global optimum  $p^*$  of (21.4). We prove that this sequence converges to the optimum when the set of constraints  $q_i(X) \geq 0$  is such that every tuple of operators  $X = (X_1, \dots, X_n)$  satisfying them are bounded, i.e., such that they satisfy  $C - (X_1 X_1^* + \dots + X_n X_n^*) \geq 0$  for some real constant  $C > 0$ . The convergence of these SDP relaxations can be shown to be related, through the duals, to an NC analogue of Putinar’s result, the Positivstellensatz for non-commutative positive polynomials introduced by Helton and McCullough [22]. We provide, however, an alternative proof of convergence through the primal that is constructive: from the sequence of optimal solutions of the SDP relaxations, we build an explicit minimizer  $(X^*, \phi^*)$ , which is defined on a (generally) infinite-dimensional Hilbert space  $H^*$ . In some cases, the SDP relaxation at a given finite step in the hierarchy may already yield the global minimum  $p^*$ . We introduce a criterion to detect when this happens, and show in this case how to extract the global minimizer  $(X^*, \phi^*)$  from the solution of this particular SDP relaxation. The corresponding Hilbert space  $H^*$  is then finite-dimensional, with its dimension determined by the rank of the matrices involved in the solution of the SDP relaxation.

Note that unconstrained NC polynomial optimization problems (i.e. the minimization of a single polynomial  $p(X)$  with no constraints of the form  $q_i(X) \geq 0$ ) have already been studied in [30]. Such problems can also be solved using SDP, as

implemented in the MATLAB toolbox [NCSOStools] [10]. Unlike constrained NC optimization (21.4), which requires a sequence of SDPs to compute the minimum, a *single* SDP is sufficient for unconstrained NC optimization [21]. This single SDP happens to correspond to the first step of our hierarchy when neglecting constraints coming from the positivity conditions  $q_i(X) \geq 0$ .

### 21.1.5 Known Applications of the Method

The general SDP approach for NC polynomial optimization described in this chapter reduces for the specific problems (21.1) and (21.2) to SDP relaxations that have been introduced somewhat earlier and already tested extensively.

The hierarchy of SDP relaxations corresponding to the problem (21.1) was originally introduced in [40] and its convergence proven in [15, 41]. These works served as a basis for the generalization to arbitrary polynomial optimization established in [49] and presented here. Since its introduction in [40], this method for characterizing the set of quantum correlations has been used to compute the maximal violation of different Bell inequalities [7, 33, 41, 44, 46, 59], always with good results. We would like to call special attention to the work of Pál and Vertesi [44, 46], who used the SDP hierarchy to bound the maximal violation of 241 different Bell inequalities. The third relaxation of the hierarchy retrieved the exact solutions up to a precision of  $10^{-8}$  in 225 cases. For the remaining 16 inequalities the gap between the upper bound and the best known lower bound is small. The convergence of the SDP hierarchy thus seems to be fast and finite for such kind of problems. This has contributed to make it the most popular algorithm in the field.

Besides these numerical results, there are more fundamental reasons that make of the SDP hierarchy a method of choice to characterize the set of quantum correlations. First, one is generally interested in characterizing quantum correlations from the “outside”, i.e., one is interested in upper-bounds to the maximization problem (21.1) (while more traditional algorithms, such as variational methods, yield lower-bounds). This is particularly important, e.g., in quantum cryptography where one wants to be sure that all possible quantum strategies available to an eavesdropper have been characterized. Second, one is often interested in maximizing a quantity over all its possible physical realizations, that is, over Hilbert spaces of arbitrary dimension. Since the SDP approach is based on algebraic constraints on the operators, it is particularly well suited to this dimension-free setting. This SDP approach is currently the unique existing method that can provide dimension-free upper-bounds of arbitrary accuracy to problems of the form (21.1). It has been used successfully to compute the winning probability of quantum non-local games [2], to design practical tests of quantum randomness [48], to prove security of quantum key distribution [35], and to understand how some features of quantum theory can be recovered from basic physical principles [1, 43].

In recent years, very successful SDPs based on “the  $N$ -representability problem” or “reduced-density-matrix methods” have been introduced in quantum chemistry to solve problems of the form (21.2) [37]. These SDP techniques can be shown to

coincide with the lowest-order relaxations of the SDP hierarchy presented here. This approach represents a very active field of research in quantum chemistry, see e.g. [19,36]. Compared to traditional electronic structure methods, the SDP approach has several advantages including its robustness and high accuracy; its main disadvantage is its computational time and memory consumption, which are still extremely large.

Note that contrarily to the problem of characterizing the possible quantum correlations between separate systems, quantum chemistry problems are not dimension-free. However, the relevant Hilbert spaces are the unique irreducible representations of algebras of operators generated by fermionic anti-commutation relations, as in (21.2). They can thus also be treated through an SDP approach based on algebraic constraints. In the case of the problem (21.2), the corresponding Hilbert space has dimension  $2^M$ , which precludes a solution through direct diagonalization; note, however, that the operator constraints in (21.2), and thus the SDP relaxations of bounded order, are only polynomial in  $M$ .

### 21.1.6 Structure of the Chapter

Sect. 21.2 contains the main results reported here. In Sects. 21.2.1 and 21.2.2, we set the notation and define the general NC problem considered here. We introduce the corresponding hierarchy of SDP relaxations in Sect. 21.2.3. In Sect. 21.2.4, we show how to detect optimality at a finite step in the hierarchy and how to extract a global optimizer, and prove the asymptotic convergence in Sect. 21.2.5. We then explain the relation with the work of Helton and McCullough in Sect. 21.2.6. We proceed by mentioning briefly how to modify the method to deal efficiently with equality constraints in Sect. 21.2.7. In Sect. 21.2.8, we illustrate our method on concrete examples. Finally, we discuss some open questions and directions for future research in Sect. 21.3.

## 21.2 Main Results

### 21.2.1 Notation and Definitions

Consider the set of  $2n$  letters  $(x_1, x_2, \dots, x_n, x_1^*, x_2^*, \dots, x_n^*)$ . One can form a *word*  $w$  of length  $|w| = k$  by joining  $k$  such letters in a particular order, e.g.:  $w = x_2 x_4^* x_3^* x_2$  or  $w = x_1 x_1 x_1$ . We will also consider the *empty word* denoted by  $1$ , whose length we take by convention to be zero ( $|1| = 0$ ). We denote by  $\mathcal{W}_d$  the set of all words of length smaller or equal to  $d$ , and by  $\mathcal{W}$  the set of all words (of unrestricted length). Given a word  $w = w_1 w_2 \dots w_k$  of length  $k$ , we define its *involution*  $w^*$  by  $w^* = w_k^* w_{k-1}^* \dots w_1^*$ , where for individual letters  $(x_i)^* = x_i^*$  and  $(x_i^*)^* = x_i$ . To simplify the notation, we will often view the  $2n$  variables  $(x_1, x_2, \dots, x_n, x_1^*, x_2^*, \dots, x_n^*)$  as  $(x_1, \dots, x_n, x_{n+1}, \dots, x_{2n})$  by identifying  $x_{i+n}$  with  $x_i^*$ , for  $i = 1, \dots, n$ .

Given these definitions, a *polynomial*  $p$  of degree  $d$  in the noncommuting variables  $(x_1, x_2, \dots, x_n, x_1^*, x_2^*, \dots, x_n^*)$  is a linear combination of words  $w \in \mathcal{W}_d$ , that is,

$$p = \sum_{|w| \leq d} p_w w, \quad (21.12)$$

where the coefficients  $p_w$  are elements of the field  $\mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}$  of real or complex numbers. Any such polynomial  $p$  can be also seen as an element of  $\mathbb{K}[x, x^*]$ , the unital free  $*$ -algebra with generators  $x = (x_1, \dots, x_n)$ . A polynomial  $p$  is said to be hermitian if  $p^* = p$ , or, in terms of its coefficients, if  $p_w^* = p_w$ .

Note that words can be interpreted as monomials and we will sometimes use the two terms interchangeably. We will then also refer to the length  $|w|$  of a word as the degree of the monomial  $w$  and to  $\mathcal{W}_d$  as a monomial basis for polynomials of degree  $d$ .

Let  $B(H)$  denote the set of bounded operators on a Hilbert space  $H$  defined on the field  $\mathbb{K}$ . Given a set of operators  $X = (X_1, X_2, \dots, X_n)$  in  $B(H)$  and a polynomial  $p(x) \in \mathbb{K}[x, x^*]$ , we define the operator  $p(X) \in B(H)$  by substituting in the expression of  $p(x)$ , 1 by  $\mathbb{I} \in B(H)$ ;  $x_i$ , by  $X_i$ ; and  $x_i^*$  or  $x_{i+n}$ , by  $X_i^*$  (the adjoint of  $X_i$ ). If  $p^* = p$  is a hermitian polynomial, then  $p(X) = p^*(X)$  is a hermitian operator and the quantity  $\langle \phi, p(X)\phi \rangle$  is real for every vector  $\phi \in H$ .

### 21.2.2 NC Polynomial Problem

Let  $p$  be a hermitian polynomial in  $\mathbb{K}[x, x^*]$ . Define similarly the polynomials  $q_i$  ( $i = 1, \dots, m_q$ ),  $r_i$  ( $i = 1, \dots, m_r$ ) and  $s_i$  ( $i = 1, \dots, m_s$ ), where  $q_i$ 's and  $s_i$ 's are hermitian. The problem that we consider is

$$\begin{aligned} p^* &= \inf_{X, \phi} \quad \langle \phi, p(X)\phi \rangle \\ \text{subject to} \quad &\|\phi\| = 1, \\ \mathbf{P}: \quad &q_i(X) \geq 0 \quad i = 1, \dots, m_q, \\ &r_i(X)\phi = 0 \quad i = 1, \dots, m_r, \\ &\langle \phi, s_i(X)\phi \rangle \geq 0 \quad i = 1, \dots, m_s, \end{aligned} \quad (21.13)$$

where the optimization is carried out over all operators  $X = (X_1, \dots, X_n)$  and normalized vectors  $\phi$  defined on Hilbert spaces  $H$  of arbitrary dimension.

In the following, we will always assume that this problem has a feasible solution, that is, that there exists a Hilbert space  $H$ , a state  $\phi \in H$  and operators  $\{X_1, \dots, X_n\} \subset B(H)$  satisfying the constraints of problem (21.13). We will also take for granted that the polynomials  $q_i(x)$  satisfy  $C - \sum_{k=1}^{2n} x_k x_k^* = \sum_i f_i(x) f_i(x)^* + \sum_{i,j} g_{ij}^*(x) q_i(x) g_{ij}(x)$ , for some  $C \in \mathbb{R}^+$  and some polynomials  $f_i(x), g_{ij}(x) \in \mathbb{K}[x, x^*]$ . We will call this condition the *Archimedean assumption*. Note that the Archimedean assumption implies that any set of operators  $(X_1, \dots, X_n)$  satisfying  $q_i(X) \geq 0$  must necessarily be bounded, i.e.,  $X_i X_i^* \leq \sum_{k=1}^{2n} X_k X_k^* \leq C$ .

### 21.2.3 The Hierarchy of SDP Relaxations

#### 21.2.3.1 Moments and Moment Matrices

Let  $(X, \phi)$  be a set of  $n$  operators and a normalized vector defining a feasible solution of problem **P**. By the Archimedean assumption, each operator  $X_i$  is necessarily bounded, and thus by the normalization of  $\phi$ , the *moments*

$$y_w = \langle \phi, w(X)\phi \rangle \in \mathbb{K}, \quad (21.14)$$

are well defined for all  $w \in \mathcal{W}$ . Note in addition that  $y_1 = \langle \phi, \mathbb{I}\phi \rangle = 1$  and that  $\sum_w s_{i,w} y_w = \sum_w s_{i,w} \langle \phi, w(X)\phi \rangle = \langle \phi, s_i(X)\phi \rangle \geq 0$ .

Given any finite set of moments  $y = (y_w)_{|w| \leq 2k} \in \mathbb{K}^{\mathcal{W}_{2k}}$ , we define the  $k^{\text{th}}$ -order *moment matrix*  $M_k(y)$  as a square matrix whose rows and columns are labeled by words  $w \in \mathcal{W}_k$  and whose  $(v, w)$ -entry is given by

$$M_k(y)(v, w) \equiv y_{v^* w} \quad (21.15)$$

for all  $|v|, |w| \leq k$ . Any moment matrix  $M_k(y)$  must be positive semidefinite. Indeed, let  $z = (z_w)_{|w| \leq k} \in \mathbb{K}^{\mathcal{W}_k}$  be an arbitrary vector. Then,

$$\begin{aligned} z^* M_k(y) z &= \sum_{v,w} z_v^* M_k(y) z_w = \sum_{v,w} z_v^* z_w y_{v^* w} \\ &= \sum_{v,w} z_v^* z_w \langle \phi, v(X)^* w(X)\phi \rangle = \langle \phi, z^*(X) z(X)\phi \rangle \geq 0, \end{aligned} \quad (21.16)$$

where  $z(X) = \sum_w z_w w(X)$ .

Given the polynomial  $q_i(x) \in \mathbb{K}[x, x^*]$  of degree  $d_i$  and the sequence of moments  $y \in \mathbb{K}^{\mathcal{W}_{2k}}$ , we define the *localizing matrix*  $M_{k-d_i}(q_i y)$  as a square matrix whose rows and columns are labeled by words  $w \in \mathcal{W}_{k-d_i}$  and whose  $(v, w)$ -entry is given by

$$M_k(q_i y)(v, w) = \sum_{|u| \leq d} q_{i,u} y_{v^* uw} \quad (21.17)$$

for all  $|v|, |w| \leq k - d_i$ . Like moment matrices, any localizing matrix must necessarily be positive semidefinite. To see why, let  $z \in \mathbb{K}^{\mathcal{W}_{k-d_i}}$  be arbitrary and compute the product  $z^* M_k(q_i y) z$ :

$$\begin{aligned} z^* M_k(q_i y) z &= \sum_{v,w} z_v^* M_k(q_i y) z_w = \sum_{u,v,w} z_v^* z_w q_{i,u} y_{v^* uw} \\ &= \sum_{v,w} z_v^* z_w \langle \phi, v(X)^* q_i(X) w(X)\phi \rangle \\ &= \langle \phi, z^*(X) q_i(X) z(X)\phi \rangle \geq 0, \end{aligned} \quad (21.18)$$

where we have used the fact that the operators  $X$  satisfy  $q_i(X) \geq 0$ .

Finally, given the polynomial  $r_i \in \mathbb{K}[x, x^*]$  of degree  $d'_i$  and the sequence of moments  $y \in \mathbb{K}^{|\mathcal{W}_{2k}|}$ , define the vector  $m_{2k-d'_i}(r_i y)$  as the vector with components indexed in  $\mathcal{W}_{2k-d'_i}$  and whose component  $w$  is equal to

$$m_{2k-d'_i}(r_i y)(w) = \sum_{|v| \leq d'_i} r_{i,v} y_{vw}. \quad (21.19)$$

We clearly have that  $m_{2k-d'_i}(r_i y) = 0$ , since

$$m_{2k-d'_i}(r_i y)(w) = \sum_{|v| \leq d'_i} r_{i,v} \langle \phi, w^*(X)v(X)\phi \rangle = \langle \phi w^*(X), r_i(X)\phi \rangle = 0. \quad (21.20)$$

### 21.2.3.2 SDP Relaxations

Let  $d_q = \max_i \deg q_i$ ,  $d_r = \max_i \deg r_i$ ,  $d_s = \max_i \deg s_i$ , and let  $2k \geq \max\{\deg p, d_q, d_r, d_s\}$ . Consider the SDP problem

$$\begin{aligned} p^k &= \min_y \sum_{|w| \leq 2k} p_w y_w \\ \text{s.t. } & y_1 = 1 \\ \mathbf{R}_k : & M_k(y) \geq 0 \\ & M_{k-d_i}(q_i y) \geq 0 \quad i = 1, \dots, m_q, \\ & m_{2k-d'_i}(r_i y) = 0 \quad i = 1, \dots, m_r, \\ & \sum_{|w| \leq 2k} s_{i,w} y_w \geq 0 \quad i = 1, \dots, m_s, \end{aligned} \quad (21.21)$$

where the optimization is performed over the set of real or complex variables  $y = (y_w)_{|w| \leq 2k} \in \mathbb{K}^{|\mathcal{W}_{2k}|}$ .

As follows from the discussion above, any sequence of moments  $y_w = \langle \phi, w(X)\phi \rangle$  generated by a normalized unitary vector  $\phi$  and a set of  $n$  operators  $X$  compatible with the constraints in  $\mathbf{P}$  constitutes a feasible solution of  $\mathbf{R}_k$ . This implies that  $p^k \leq p^\star$ , i.e., the SDP problem  $\mathbf{R}_k$  is a relaxation of  $\mathbf{P}$ . Notice further that any feasible solution  $y = (y_w)_{|w| \leq 2(k+1)}$  of  $\mathbf{R}_{k+1}$  provides a feasible solution of  $\mathbf{R}_k$  by discarding all  $y_w$  with  $|w| = 2k+1, 2k+2$ . This implies that  $\mathbf{R}_k$  is itself a relaxation of  $\mathbf{R}_{k+1}$  and thus that the sequence of optima  $p^k$  is a monotonically increasing sequence:  $p^k \leq p^{k+1} \leq p^{k+2} \dots \leq p^\star$ .

### 21.2.4 Optimality Detection

The optimal solution of a NC problem may be attained at a finite relaxation step [41]. We now prove a result that permits to detect whether the relaxation  $\mathbf{R}_k$  provides the optimal value  $p^*$ .

**Theorem 21.1.** *Let  $y \in \mathbb{K}^{|\mathcal{W}^{2k}|}$  be an optimal solution of the relaxation of order  $k$  such that*

$$\text{rank } M_k(y) = \text{rank } M_{k-d}(y), \quad (21.22)$$

and

$$d' - d \leq k, \quad (21.23)$$

where  $d = \max_i \deg q_i \geq 1$  and  $d' = \max_i \deg r_i \geq 0$ . Then  $p^k = p^*$ , i.e., the optimum of the relaxation of order  $k$  is the global optimum of the original problem  $\mathbf{P}$ . Moreover, one can construct from  $y$  an explicit global optimizer  $(X, \phi)$  for the problem  $\mathbf{P}$  with  $\dim H = \text{rank } M_{k-d}(y)$ .

The proof of this theorem is based on the following lemma which can be viewed as a non-commutative analogue of the flat extension theorem of Curto and Fialkow [14] for the commutative case.

**Lemma 21.1.** *Let  $y \in \mathbb{K}^{|\mathcal{W}^{2k}|}$  be such that  $M_k(y) \geq 0$  and  $\text{rank } M_k(y) = \text{rank } M_{k-d}(y)$ , with  $d \geq 1$ . Then  $y$  can be interpreted as a sequence of moments  $y_w = \langle \phi, w(X) \phi \rangle$  for some  $(X, \phi)$ . Moreover, the operators  $X$  and the vector  $\phi$  are defined on a Hilbert space  $H$  of dimension  $\dim(H) = \text{rank } M_{k-d}(y)$  spanned by the vectors  $\{w(X)\phi : w \in \mathcal{W}_{k-d}\}$ .*

*Proof.* Let  $r = \text{rank } M_k(y)$ . Since  $M_k(y)$  is positive semidefinite, it admits a Gram decomposition [25], i.e., for each word  $w \in \mathcal{W}_k$  there exists a vector  $\bar{w} \in \mathbb{K}^r$  such that

$$M_k(y)(v, w) = y_{v^* w} = \langle \bar{v}, \bar{w} \rangle. \quad (21.24)$$

Now, the rank of any  $s \times s$  matrix  $A$  of the form  $A_{ij} = \langle \bar{v}_i, \bar{v}_j \rangle$  is equal to the number of linearly independent vectors in  $\{\bar{v}_i : i = 1, \dots, s\}$ <sup>1</sup>. The condition that  $\text{rank } M_k(y) = \text{rank } M_{k-d}(y)$  is therefore equivalent to

$$\text{span}\{\bar{w} : |w| \leq k\} = \text{span}\{\bar{w} : |w| \leq k - d\}. \quad (21.25)$$

And, in particular, it implies that

$$\text{span}\{\bar{w} : |w| \leq k\} = \text{span}\{\bar{w} : |w| \leq k - 1\}. \quad (21.26)$$

---

<sup>1</sup>To see this, take  $\{\bar{u}_i\}$  to be the canonical basis of  $\mathbb{K}^s$ , and note that  $A = B^* B$ , with  $B = \sum_{i=1}^s \bar{v}_i \cdot (\bar{u}_i)^*$ . It follows that  $\text{rank}(A) = \text{rank}(B) = \dim(\text{span}\{\bar{v}_i\}_i)$ .

Let  $H$  be the Hilbert space  $H = \text{span}\{\bar{w} : |w| \leq k\}$  of dimension  $\dim(H) = r$ . Now, introduce  $2n$  operators  $X_i$  and define their action over the vectors  $\{\bar{w} : |w| \leq k-1\}$  through

$$X_i \bar{w} \equiv \overline{x_i w}, \quad \text{for } |w| \leq k-1. \quad (21.27)$$

The action of  $X_i$  over an arbitrary vector  $\bar{f} = \sum_{|w| \leq k-1} a_w \bar{w} \in H$  is given by  $X_i \bar{f} \equiv \sum_{|w| \leq k-1} a_w \overline{x_i w}$ . To see that this definition is consistent, notice that, for any two different decompositions of the same vector  $\bar{f} = \sum_{|w| \leq k-1} a_w \bar{w} = \sum_{|w| \leq k-1} b_w \bar{w}$ , we have that  $\sum_{|w| \leq k-1} a_w \overline{x_i w} = \sum_{|w| \leq k-1} b_w \overline{x_i w}$ . Indeed, for any  $v \in \mathcal{W}_{k-1}$ ,

$$\begin{aligned} \left\langle \bar{v}, \sum_w (a_w - b_w) \overline{x_i w} \right\rangle &= \sum_w (a_w - b_w) y_{v^* x_i w} = \sum_w (a_w - b_w) y_{(x_i^* v)^* w} \\ &= \left\langle \overline{x_i^* v}, \sum_w (a_w - b_w) \bar{w} \right\rangle = \langle \overline{x_i^* v}, 0 \rangle = 0. \end{aligned} \quad (21.28)$$

Since the vectors  $\{\bar{v} : |v| \leq k-1\}$  span the whole space  $H$ , we conclude that  $\sum_{|w| \leq k-1} a_w \overline{x_i w} - \sum_{|w| \leq k-1} b_w \overline{x_i w} = 0$ , and thus the operators  $X_i$  are well defined over  $H$ .

These operators are also consistent with respect to the involution  $*$  on  $\mathbb{K}[x, x^*]$ , i.e.,  $X_i^* = X_{i+n}$  for all  $i = 1, \dots, n$ . Indeed, for any pair of words  $v, w \in \mathcal{W}_{k-1}$ ,

$$\langle \bar{v}, X_i^* \bar{w} \rangle = \langle X_i \bar{v}, \bar{w} \rangle = \langle \overline{x_i v}, \bar{w} \rangle = y_{v^* x_i^* w} = y_{v^* x_{i+n} w} = \langle \bar{v}, X_{i+n} \bar{w} \rangle. \quad (21.29)$$

Define now  $\phi \equiv \bar{1} \in H$ . Then, for any pair of words  $|v|, |w| \leq k$  we have that

$$\langle \phi, v(X)w(X)\phi \rangle = \langle v^*(X)\phi, w(X)\phi \rangle = \langle \overline{v^*}, \bar{w} \rangle = y_{vw}, \quad (21.30)$$

which implies that for any  $|w| \leq 2k$ ,  $y_w = \langle \phi, w(X)\phi \rangle$ . Finally, note that, due to (21.25),  $H = \text{span}\{\bar{w} : |w| \leq k-d\} = \text{span}\{w(X)\phi : w \in \mathcal{W}_{k-d}\}$ .  $\square$

*Proof (of Theorem 21.1).* Let  $y$  be as in the hypothesis of Theorem 21.1. By Lemma 21.1, there exist  $2n$  operators  $X$  and a state  $\phi$  in a Hilbert space of dimension  $\dim H = \text{rank } M_{k-d}(y)$ , spanned by the vectors  $\{w(X)\phi : w \in \mathcal{W}_{k-d}\}$ , and such that  $y_w = \langle \phi, w(X)\phi \rangle$ . We now show that  $(X, \phi)$  is a feasible solution for  $\mathbf{P}$ , i.e., that  $\phi$  is normalized and that  $q_i(X) \geq 0$ ,  $r_i(X)\phi = 0$ , and  $\langle \phi, s_i(X)\phi \rangle \geq 0$  for all  $i$ . This implies that  $p^* \leq \langle \phi, p(X)\phi \rangle = p^k$ . Since we also have that  $p^k \leq p^*$ , it follows that  $p^k = p^*$ .

First, observe that the normalization of  $\phi$  follows from the condition  $y_1 = 1$ , and that  $\langle \phi, s_i(X)\phi \rangle = \sum_w s_{i,w} y_w \geq 0$ . To show that  $q_i(X) \geq 0$ , use the fact that  $H = \text{span}\{\bar{w} : |w| \leq k-d\}$ , where  $\bar{w} = w(X)\phi$  as implied by Lemma 21.1. Showing that  $q_i(X) \geq 0$  is thus equivalent to prove that, for any vector  $c \in \mathbb{K}^{|\mathcal{W}_{k-d}|}$ ,  $\langle \sum_v c_v^* \bar{v}, q_i(X) \sum_w c_w \bar{w} \rangle \geq 0$ , where the sums run over  $|v|, |w| \leq k-d$ . But the left hand side of this inequality is equal to  $\sum_{u,v,w} c_v^* q_u y_{v^* u w} c_w = c^* M_{k-d}(q_i y) c$ , which is non-negative since  $M_{k-d}(q_i y) \geq 0$ . Similarly,  $r_i(X)\phi = 0$  follows from the fact that  $\langle \bar{v}, r_i(X)\phi \rangle = \sum_w r_w y_{v^* w} = 0$  for all  $|v| \leq k-d$  provided that  $2k - d'_i \geq k-d$ , i.e., when (21.23) is satisfied.  $\square$

### 21.2.5 Asymptotic Convergence

The convergence of the hierarchy at a finite order  $k$  cannot be guaranteed in general. Can we say something about its asymptotic convergence? Note that the limit  $\lim_{k \rightarrow \infty} p^k = \hat{p}$  always exists, since the sequence of optima  $p^k$  of the relaxations  $\mathbf{R}_k$  is a monotonically increasing sequence which is bounded from above by  $p^*$ . We now show that  $\hat{p} = p^*$ .

**Theorem 21.2.** *Let the polynomials  $q_i(x)$  ( $i = 1, \dots, m_q$ ) satisfy the Arquimedean assumption. Then  $\lim_{k \rightarrow \infty} p^k = p^*$ .*

Let  $y = (y_w)_{w \in \mathcal{W}} \in \mathbb{K}^\infty$  be an infinite sequence of elements of  $\mathbb{K}$ , indexed by the set of all words  $\mathcal{W}$ . We write  $M(y) \geq 0$  if  $M_k(y) \geq 0$  for all  $k \geq 1$ , and  $m(y) = 0$  if  $m_{2k-d'_i}(r_i y) = 0$  for all  $k \geq d'_i/2$ .

The proof of Theorem 21.2 uses the following lemma, proven in [49].

**Lemma 21.2.** *Let the polynomials  $q_i(x)$  ( $i = 1, \dots, m_q$ ) satisfy the Arquimedean assumption and let  $\hat{p} = \lim_{k \rightarrow \infty} p^k$  be the limit of the optimal solutions  $p^k$  of the relaxations  $\mathbf{R}_k$ . Then there exists in  $\mathbb{K}$  an infinite sequence  $y$  such that  $|y_w| \leq C^{|w|}$ , for all  $w$ , and*

$$\sum_w p_w y_w = \hat{p}, \quad (21.31)$$

$$y_1 = 1, \quad (21.32)$$

$$M(y) \geq 0, \quad (21.33)$$

$$M(q_i y) \geq 0 \quad i = 1, \dots, m, \quad (21.34)$$

$$m(r_i y) = 0, \quad (21.35)$$

$$\sum_w s_{i,w} y_w \geq 0. \quad (21.36)$$

We now proceed to the proof of Theorem 21.2.

*Proof (of Theorem 21.2).* The principle of the proof is the same as that of Theorem 21.1: we show that there exists an optimizer<sup>2</sup>  $(H, X, \phi)$  such that  $\langle \phi, p(X)\phi \rangle = \lim_{k \rightarrow \infty} p^k$ . In this way, we also prove that the infimum appearing in (21.13) is attainable, i.e., it is actually a minimum.

Let  $\hat{p} = \lim_{k \rightarrow \infty} p^k$  and let  $y$  be the infinite sequence defined in Lemma 21.2. We start by showing that given  $y$  one can build a set of vectors  $V = \{\bar{w} : w \in \mathcal{W}\}$  such that  $\langle \bar{v}, \bar{w} \rangle = y_{v^* w}$ .

---

<sup>2</sup>The proof that we give is similar to the one of Lemma 21.1. An alternative proof based on a Gelfand–Naimark–Segal like construction is given in [49].

Let  $r = \lim_{k \rightarrow \infty} \text{rank } M_k(y)$  (if this limit does not exist, take  $r = \infty$ ). Choose  $k \geq \lceil \max\{\deg(p), d_q, d_r, d_s\}/2 \rceil$  and perform a Gram decomposition of the positive semidefinite matrix  $M_k(y)$ , i.e., find vectors  $V_k = \{\bar{w} \in l_2'(\mathbb{K}) : |w| \leq k\}$  such that  $\langle \bar{v}, \bar{w} \rangle = y_{v^* w}$ , for all  $|v|, |w| \leq k$ . These will be the first  $|\mathcal{W}_k|$  vectors of the set  $V$ . We now show that this set can be extended to a larger set of vectors  $V_{k+1} = V_k \cup \{\bar{w} : |w| = k+1\}$  such that  $\langle \bar{v}, \bar{w} \rangle = y_{v^* w}$  for all  $|v|, |w| \leq k+1$ . By iterating this procedure one obtains a well-defined set  $V = \{\bar{w} : w \in \mathcal{W}\}$  such that  $\langle \bar{v}, \bar{w} \rangle = y_{v^* w}$  for all  $v, w$ .

To define the new vectors  $\{\bar{w} : |w| = k+1\}$ , we perform a Gram decomposition of the matrix  $M_{k+1}(y)$ , thus obtaining a collection of vectors  $\{\bar{w}' \in l_2'(\mathbb{K}) : |w| \leq k+1\}$ . Notice that, for any  $|v|, |w| \leq k$ ,  $\langle \bar{v}, \bar{w} \rangle = \langle \bar{v}', \bar{w}' \rangle = y_{v^* w}$ , i.e., the scalar products of both sets of vectors are the same for words  $v, w$  of length up to  $k$ . This implies that there exists a unitary (or an orthogonal) transformation  $U$  such that  $\bar{w} = U\bar{w}'$ , for all  $|w| \leq k$ . Choosing  $V_{k+1}$  to be  $\{\bar{w} : |w| \leq k\} \cup \{U\bar{w}' : |w| = k+1\}$ , we have that  $V_{k+1}$  is of the desired form  $V_{k+1} = V_k \cup \{\bar{w} : |w| = k+1\}$  and that  $\langle \bar{v}, \bar{w} \rangle = \langle \bar{v}', \bar{w}' \rangle = y_{v^* w}$ , for  $|v|, |w| \leq k+1$ .

Given a set of vectors  $V = \{\bar{w} : w \in \mathcal{W}\}$  such that  $\langle \bar{v}, \bar{w} \rangle = y_{v^* w}$ , we define  $2n$  operators through their action over the elements of  $V$  as in the proof of Lemma 21.1 (21.27). By the same arguments, this definition can be linearly extended to the pre-Hilbert space  $H = \text{span}\{\bar{w} : w \in \mathcal{W}\}$  (note that  $X_i a \in H$  for all  $a \in H$ ), with the operators  $X_i$  also satisfying  $\langle a, X_i^* b \rangle = \langle a, X_{i+n} b \rangle$  for any  $a, b \in H$ . Likewise, taking  $\phi = \bar{1}$ , one can show that  $\langle \phi, w(X)\phi \rangle = y_w$ , for all  $w \in \mathcal{W}$ . Similarly to the proof of Theorem 21.1, then it follows from the conditions (21.32), (21.34), (21.35), and (21.36) that

$$\begin{aligned} \langle a, q_i(X)a \rangle &\geq 0 \quad \text{for any } a \in H, \\ \langle \phi, s_i(X)\phi \rangle &\geq 0, \\ \langle a, r_i(X)\phi \rangle &= 0 \quad \text{for all } a \in H. \end{aligned} \tag{21.37}$$

It only remains to see that the action of the operators  $X_i$  is also well defined over the closure of  $H$ . Remember that the Arquimedean assumption states that

$$C - \sum_i x_i^* x_i = \sum_j f_j^*(x) f_j(x) + \sum_{i,j} g_{ij}^*(x) q_i(x) g_{ij}(x), \tag{21.38}$$

for some polynomials  $f_j(x), g_{ij}(x)$ . For any  $l \in \{1, \dots, 2n\}$  and any vector  $a \in H$ , this implies that

$$\begin{aligned} C\langle a, a \rangle - \langle X_l a, X_l a \rangle &\geq C\langle a, a \rangle - \sum_i \langle X_i a, X_i a \rangle \\ &= \sum_{i,j} \langle g_{ij}(X)a, q_i(X)g_{ij}(X)a \rangle + \sum_j \langle f_j(X)a, f_j(X)a \rangle \geq 0. \end{aligned} \tag{21.39}$$

Now, let  $(a_n), (b_n)$  be two sequences of vectors in  $H$  such that  $\lim_{n,m \rightarrow \infty} \|a_n - b_m\| = 0$ . From (21.39) it then follows that

$$\lim_{n,m \rightarrow \infty} \|X_i a_n - X_i b_m\| \leq \lim_{n,m \rightarrow \infty} \sqrt{C} \|a_n - b_m\| = 0. \quad (21.40)$$

The action of  $X_i$  can therefore be extended to the closure of  $H$  by defining the action of  $X_i$  over sequences of elements of  $H$  in the usual way.

### 21.2.6 Duals and the Positivstellensatz for NC Polynomials

The duals of the relaxations  $\mathbf{R}_k$  can be shown to be equivalent to the problems

$$\begin{aligned} \lambda^k &= \max_{\lambda, b_i, c_{ij}, f_i, g_i} \lambda \\ \text{s.t. } p - \lambda &= \sum_j b_j^* b_j + \sum_{i=1}^{m_q} \sum_j c_{ij}^* q_i c_{ij} \\ &\quad + \sum_{i=1}^{m_r} (f_i r_i + r_i^* f_i^*) + \sum_{i=1}^{m_s} g_i s_i \\ &\quad \max_j \deg(b_j) \leq k, \\ &\quad \max_j \deg(c_{ij}) \leq k - d_i, \\ &\quad \deg(f_i) \leq 2k - d'_i, \\ &\quad g_i \geq 0, \end{aligned} \quad (21.41)$$

where  $b_j, c_{ij}, f_i$  are polynomials and  $g_i$  are real numbers. From the decomposition of  $p - \lambda^k$  appearing in (21.41), it clearly follows that  $p(X) - \lambda^k \geq 0$  for any  $(X, \phi)$  satisfying the constraints in  $\mathbf{P}$ . Thus the solution of the dual (21.41) provides a certificate that the optimal solution  $p^*$  of  $\mathbf{P}$  cannot be lower than  $\lambda^k$ . More generally, since (21.41) is the dual of (21.21), we have  $\lambda^k \leq p^k$ .

Consider the case where the polynomials  $r_i$  and  $s_i$  do not appear in (21.41), corresponding to the simplified problem

$$\begin{aligned} \mathbf{P}: \quad p^* &= \inf_{X, \phi} \langle \phi, p(X)\phi \rangle \\ \text{subject to } q_i(X) &\geq 0 \quad i = 1, \dots, m, \end{aligned} \quad (21.42)$$

that does not involve constraints of the form  $r_i(X)\phi = 0$  and  $\langle \phi, s_i(X)\phi \rangle \geq 0$ . In this case the asymptotic convergence of the hierarchy of relaxations can be proven, alternatively to Theorem 21.2, using the remarkable results of Helton and McCullough [22] in the field of sum of squares (SOS) decompositions (see [15, 49]).

Indeed, the relaxations (21.41) then correspond to seeking SOS decomposition for  $p$  of the form

$$p - \lambda^k = \sum_j b_j^* b_j + \sum_i \sum_j c_{ij}^* q_i c_{ij}, \quad (21.43)$$

in terms of polynomials  $b_j$  and  $c_{ij}$  of bounded degree in  $k$ . From the definition of problem **P**, it is clear that  $p(X) - p^* + \epsilon > 0$ , for all  $\epsilon > 0$  and all operators  $X$  satisfying  $q_i(X) \geq 0$ . The Positivstellensatz representation theorem of Helton and McCullough [22] then tells us that the polynomial  $p - p^* + \epsilon$  necessarily admits a decomposition as in the right-hand side of (21.43) for some polynomials  $b_j, c_{ij}$  of undefined degree. This implies that  $\lim_{k \rightarrow \infty} \lambda^k \geq p^* - \epsilon$ . From the chain of inequalities  $\lambda^k \leq p^k \leq p^*$  and the fact that  $\epsilon$  is arbitrary it follows that  $\lim_{k \rightarrow \infty} \lambda^k = \lim_{k \rightarrow \infty} p^k = p^*$ .

### 21.2.7 Equality Constraints

The problem **P** may contain constraints that appear in the form of equalities  $q(X) = 0$ , rather than inequalities  $q(X) \geq 0$ . These cases can also be treated in the above formalism by defining two localizing matrices  $M_{k-d}(qy) \geq 0, M(-qy) \geq 0$  instead of one. Since both matrix inequalities imply that  $M_{k-d}(qy) = M_{k-d}(-qy) = 0$ , it is more efficient from a computation point of view to replace them by the linear constraints

$$\sum_u q_{uy} y_{v^*uw} = 0 \quad \text{for all } |v|, |w| \leq k-d, \quad (21.44)$$

on the variables  $y_w$ .

A cleverer way to deal with equalities, though, is to express every polynomial  $p$  modulo the ideal generated by the set of equality constraints, as indicated in [49]. Roughly speaking, given a finite set of equalities  $E = \{e_i(x) = 0\}_i$ , let  $I = \{\sum_i f_i e_i g_i : f_i, g_i \in \mathbb{K}[x, x^*]\}$  be the ideal generated by these inequalities. The idea is to find a monomial basis  $\mathcal{B}$  for the quotient ring  $\mathbb{K}[x, x^*]/I$  such that any polynomial  $p \in \mathbb{K}[x, x^*]$  can be decomposed as  $p = \sum_{w \in \mathcal{B}} q_w w + I$  for a unique  $q = \sum_{w \in \mathcal{B}} q_w w$ . All the results presented here still hold if we work in  $\mathbb{K}[x, x^*]/I$  and build the relaxations from the monomial basis  $\mathcal{B}$ .

All the problem of course consists in building a monomial basis  $\mathcal{B}$  for the quotient ring  $\mathbb{K}[x, x^*]/I$ . This can be done, e.g., if a finite Gröbner basis exists and can be computed efficiently for the ideal  $I$  [38]. Most physical problems involve equality constraints for which such a reduced monomial basis  $B$  is readily obtained (e.g.,  $x_i^* = x_i, x_i x_j = -x_j x_i, x_i^2 = x_i, \dots$ ).

The case of hermitian variables is of particular interest. All the results presented so far carry over to this situation by viewing polynomials as elements of the free  $*$ -algebra  $\mathbb{K}[x]$  with generators  $x = (x_1, \dots, x_n)$  and anti-involution  $*$  defined on letters as  $x_i^* = x_i$  (i.e., the effect of the involution operation  $*$  over any word is simply to invert the order of its letters). Another case of interest is the one of commuting variables  $x_i x_j = x_j x_i$  for which the SDP relaxations presented here reduce to those introduced by Lasserre [31].

To illustrate the reduction in complexity that follows from working in the quotient ring  $\mathbb{K}[x, x^*]/I$ , consider the case of hermitian variables satisfying  $x_i^* = x_i$  and  $x_i^2 = x_i$ . Then the appropriate basis  $\mathcal{B}$  is the set of all words that result from concatenating letters of the alphabet  $\{x_1, x_2, \dots, x_n\}$  in such a way that no two consecutive letters coincide. The effect of the involution  $*$  is simply to invert the order of the word, but composition rules would experience a small change: the concatenation of any two monomials of the form  $w_1 = \tilde{w}_1 x_k, w_2 = x_k \tilde{w}_2$  should be identified with the monomial  $\tilde{w}_1 x_k \tilde{w}_2$  instead of  $\tilde{w}_1 x_k^2 \tilde{w}_2$ . For instance, if  $n = 2$  then the rows and columns of the moment matrix  $M_2(y)$  are labeled by the monomials  $\mathcal{B}_2 = \{1, x_1, x_2, x_1 x_2, x_2 x_1\}$ , i.e.,  $M_2(y)$  is a  $5 \times 5$  matrix. The original relaxation in the basis  $\mathcal{W}_2$ , however, involves a  $21 \times 21$  matrix. The comparison becomes more advantageous as  $k$  grows: the size  $s$  of the moment matrix indexed in  $\mathcal{B}_k$  is linear in  $k$  ( $s = 1 + 2k$ ), while it scales exponentially ( $s = (4^{k+1} - 1)/3$ ) for the original relaxation in  $\mathcal{W}_k$ .

### 21.2.8 Illustration of the Method on a Specific Example

For the sake of clarity, we now apply our approach to a particular problem involving hermitian variables. Suppose, then, that we want to solve the NC polynomial optimization problem

$$\begin{aligned} p^* &= \min_{X, \phi} \langle \phi, X_1 X_2 + X_2 X_1 \phi \rangle \\ \text{s.t. } &X_1^2 - X_1 = 0 \\ &-X_2^2 + X_2 + 1/2 \geq 0. \end{aligned} \tag{21.45}$$

Let us first notice that

$$3 - x_1^2 - x_2^2 = (1 - x_1)^2 + (1 - x_2)^2 + 2(x_1 - x_1^2) + 2(-x_2^2 - x_2 + 1/2). \tag{21.46}$$

Thus the Archimedean assumption holds, and so the sequence of relaxations converges to the solution of the problem.

Due to the equality constraints  $x_1^2 = x_1$ , the appropriate monomial basis  $\mathcal{B}$  to attack this problem is the set of finite sequences of characters  $\{x_1, x_2\}$  where the letter  $x_1$  does not appear twice consecutively. Since all constraint and objective variables are at most of degree 2, the first relaxation step corresponds to the case  $k = 1$ , and is associated with the monomial basis  $\mathcal{B}_2 = \{1, x_1, x_2, x_1 x_2, x_2 x_1, x_2^2\}$ . To simplify the notation, let us denote the variables  $y_1, y_{x_1}, y_{x_2}, y_{x_1 x_2}, \dots$  as  $y_0, y_1, y_2, y_{12}, y_{21}, y_{22}\}$  and corresponds to the SDP problem

$$\begin{aligned}
p^1 = \min_y & y_{12} + y_{21} \\
\text{s.t. } & \left[ \begin{array}{c|cc} 1 & y_1 & y_2 \\ \hline y_1 & y_1 & y_{12} \\ y_2 & y_{21} & y_{22} \end{array} \right] \geq 0 \\
& -y_{22} + y_2 + 1/2 \geq 0.
\end{aligned} \tag{21.47}$$

We solved this SDP problem using the Matlab toolboxes YALMIP [34] and SeDuMi [56]. After rounding, we obtain the solution  $p^1 = -3/4$ , achieved for the moment matrix

$$M_1 = \left[ \begin{array}{c|cc} 1 & 3/4 & -1/4 \\ \hline 3/4 & 3/4 & -3/8 \\ -1/4 & -3/8 & 1/4 \end{array} \right], \tag{21.48}$$

with eigenvalues  $0, 1 \pm \sqrt{37}/8$ . The second order relaxation is

$$\begin{aligned}
p^2 = \min_y & y_{12} + y_{21} \\
\text{s.t. } & \left[ \begin{array}{c|ccc|ccc} 1 & y_1 & y_2 & y_{12} & y_{21} & y_{22} \\ \hline y_1 & y_1 & y_{12} & y_{12} & y_{121} & y_{122} \\ y_2 & y_{21} & y_{22} & y_{212} & y_{221} & y_{222} \\ \hline y_{21} & y_{21} & y_{212} & y_{212} & y_{2121} & y_{2122} \\ y_{12} & y_{121} & y_{122} & y_{1212} & y_{1221} & y_{1222} \\ y_{22} & y_{221} & y_{222} & y_{2212} & y_{2221} & y_{2222} \end{array} \right] \geq 0 \\
& \left[ \begin{array}{c|c} -y_{22} + y_2 + \frac{1}{2} & -y_{221} + y_{21} + \frac{1}{2}y_1 \\ \hline -y_{221} + y_{21} + \frac{1}{2}y_1 & -y_{1221} + y_{121} + \frac{1}{2}y_1 \\ -y_{222} + y_{22} + \frac{1}{2}y_2 & -y_{1222} + y_{122} + \frac{1}{2}y_{12} \\ \hline & \begin{array}{c} -y_{222} + y_{22} + \frac{1}{2}y_2 \\ -y_{1222} + y_{122} + \frac{1}{2}y_{12} \\ -y_{2222} + y_{222} + \frac{1}{2}y_{22} \end{array} \end{array} \right] \geq 0,
\end{aligned} \tag{21.49}$$

with solution  $p^2 = -3/4$ . The moment matrix associated to this solution is

$$M_2 = \left[ \begin{array}{c|ccc|ccc} 1 & 3/4 & -1/4 & -3/8 & -3/8 & 1/4 \\ \hline 3/4 & 3/4 & -3/8 & -3/8 & -3/16 & 0 \\ -1/4 & -3/8 & 1/4 & 3/16 & 0 & 1/8 \\ \hline -3/8 & -3/8 & 3/16 & 3/16 & 3/32 & 0 \\ -3/8 & -3/16 & 0 & 3/32 & 3/16 & -3/16 \\ 1/4 & 0 & 1/8 & 0 & -3/16 & 1/4 \end{array} \right], \tag{21.50}$$

which has two non-zero eigenvalues  $3/32 \times (14 \pm \sqrt{61})$ .

### 21.2.8.1 Optimality Criterion and Extraction of Optimizers

Since the matrix  $M_2$  has two non-zero eigenvalues, it has rank 2. Let  $M_1(y^2)$  be the upper-right  $3 \times 3$  submatrix of  $M_2 = M_2(y^2)$ . This submatrix is, in fact, equal to (21.48) and thus also has rank 2. The matrices  $M_1(y^2)$  and  $M_2(y^2)$  have then the same rank and the condition (21.22) of Theorem 21.1 is satisfied. It follows that  $p^* = p^2 = -3/4$ . It also follows that we can extract a global optimizer for (21.47), which will be realized in a space of dimension 2. For this, write down the Gram decomposition  $M_2 = R^T R$ , where

$$R = \begin{bmatrix} 1 & 3/4 & -1/4 & -3/8 & -3/8 & 1/4 \\ 0 & \sqrt{3}/4 & -\sqrt{3}/4 & \sqrt{3}/8 & \sqrt{3}/8 & -\sqrt{3}/4 \end{bmatrix}. \quad (21.51)$$

Following the procedure specified in the proof of Theorem 21.1 and Lemma 21.1, we find the optimal solutions

$$X_1^* = \begin{bmatrix} 3/4 & \sqrt{3}/4 \\ \sqrt{3}/4 & 1/4 \end{bmatrix}, \quad X_2^* = \begin{bmatrix} -1/4 & -\sqrt{3}/4 \\ -\sqrt{3}/4 & 5/4 \end{bmatrix}, \quad \phi^* = \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \quad (21.52)$$

### 21.2.8.2 Dual

Solving the dual of the order 1 relaxation (21.47) yields the SOS decomposition

$$x_1 x_2 + x_2 x_1 - \left( -\frac{3}{4} \right) = \left( -\frac{1}{2} + x_1 + x_2 \right)^2 + \left( -x_2^2 + x_2 + \frac{1}{2} \right). \quad (21.53)$$

It immediately follows that  $\langle \phi, X_1 X_2 + X_2 X_1 \phi \rangle \geq -3/4$  for every  $(X, \phi)$  satisfying  $X_1^2 = X_1$  and  $-X_2^2 + X_2 + \frac{1}{2} \geq 0$ . Thus the decomposition (21.53) provides a certificate that the solution (21.52) is optimal.

### 21.2.8.3 Comparison with the Commutative Case

To illustrate the differences and similarities between the non-commutative and the commutative case, let

$$\begin{aligned} p^* &= \min_{x \in \mathbb{R}^2} 2x_1 x_2 \\ \text{s.t. } &x_1^2 - x_1 = 0 \\ &-x_2^2 + x_2 + 1/2 \geq 0 \end{aligned} \quad (21.54)$$

be the commutative version of (21.45). The first relaxation step associated to this problem involves the monomial basis  $\mathcal{B}_2^c = \{1, x_1, x_2, x_1 x_2, x_2^2\}$  (we used  $x_1 x_2 = x_2 x_1$ )

and the corresponding relaxation variables  $\{y_0, y_1, y_2, y_{12}, y_{22}\}$ . This should be compared to  $\mathcal{B}_2 = \{1, x_1, x_2, x_1 x_2, x_2 x_1 x_2^2\}$  and  $\{y_0, y_1, y_2, y_{12}, y_{21}, y_{22}\}$  in the non-commutative case. The first order relaxation associated to (21.54) is thus

$$\begin{aligned} p^1 &= \min_y 2y_{12} \\ \text{s.t. } &\left[ \begin{array}{c|cc} 1 & y_1 & y_2 \\ \hline y_1 & y_1 & y_{12} \\ y_2 & y_{12} & y_{22} \end{array} \right] \geq 0 \\ &-y_{22} + y_2 + 1/2 \geq 0. \end{aligned} \quad (21.55)$$

Note that (21.47) and (21.55) are in fact identical, because the hermiticity of the moment matrix in (21.47) implies that  $y_{12} = y_{21}$ . Actually, it always happens that the first order relaxations of the commutative and non-commutative version of a problem coincide. We thus find as before that  $p^1 = -3/4$ . The relaxation of order two of (21.55), however, is

$$\begin{aligned} p^2 &= \min_y 2y_{12} \\ \text{s.t. } &\left[ \begin{array}{c|cc|cc} 1 & y_1 & y_2 & y_{12} & y_{22} \\ \hline y_1 & y_1 & y_{12} & y_{12} & y_{122} \\ y_2 & y_{12} & y_{22} & y_{122} & y_{222} \\ \hline y_{12} & y_{12} & y_{122} & y_{122} & y_{1222} \\ y_{22} & y_{122} & y_{222} & y_{1222} & y_{2222} \end{array} \right] \geq 0 \\ &\left[ \begin{array}{c|cc} -y_{22} + y_2 + \frac{1}{2} & -y_{122} + y_{12} + \frac{1}{2}y_1 \\ \hline -y_{122} + y_{12} + \frac{1}{2}y_1 & -y_{122} + y_{12} + \frac{1}{2}y_1 \\ -y_{222} + y_{22} + \frac{1}{2}y_2 & -y_{1222} + y_{122} + \frac{1}{2}y_{12} \\ & -y_{2222} + y_{222} + \frac{1}{2}y_{22} \end{array} \right] \geq 0. \end{aligned} \quad (21.56)$$

Solving it, we obtain  $p^2 = 1 - \sqrt{3} \approx -0.7321$ . Again, it can be verified that the rank condition (21.22) of Theorem 21.1 is satisfied, so that this solution is optimal, and the following optimizer can be reconstructed:

$$x_1^\star = 1, \quad x_2^\star = (1 - \sqrt{3})/2. \quad (21.57)$$

As expected, the global minimum of (21.54) is higher than the one of (21.45) as the commutative case is more constrained than the non-commutative one.

## 21.3 Discussion and Open Questions

This last section has a more speculative character and discusses directions for future research. We present in particular several open questions where the NC formalism that we just presented may find an application. Most of the problems have a quantum origin, due to the intrinsic non-commutativity of quantum theory, but we also provide examples with no quantum motivation.

We would like to mention as well that the emerging new field of free probability theory [60] and the theory of non-commutative operator spaces [50] are using rather intensively positivity results and finite rank approximations. Links between these two fields and the topics discussed here may be expected in the near future. Actually, in the last years operator-space theory has successfully been applied to the study of quantum correlations [28], one of the original motivations for the present work.

### 21.3.1 Extending Results in the Commutative Case to the NC Setting

As we noted earlier, when we require that the variables  $x$  commute, the SDP hierarchy reduces to the one introduced by Lasserre. The commutative setting has generated quite a large literature and the properties of the corresponding SDP relaxations have been thoroughly investigated. We refer to [32] for a review. This paper provides an NC analogue only of the most basic results in the commutative case. It would be interesting to reexamine from an NC perspective other topics in this subject, such as exploiting sparsity and symmetries, bounds on the speed of convergence, etc.

### 21.3.2 Finite Versus Infinite Dimension

Many optimization problems involve operators (or matrices) defined on a Hilbert space of known and finite dimension. How can our construction be modified to deal with these cases? A rather straightforward solution to deal with such dimension-dependent problems consists in introducing a basis of operators and express all the operators appearing in the original optimization problem in this basis. The resulting problem now involves only scalar, commuting variables (the components of each operator in this basis) and this leads to a set of polynomial constraints that can be solved using standard SDP relaxations [24, 31]. This solution however becomes impractical already for relatively small dimension.

On the other hand, our approach is insensitive to these scaling problems since each operator is treated as a single variable. However, it does not take advantage

of any constraint imposed by the known finite dimension. It would be interesting to understand how to incorporate information about the dimension in the present formalism. One possibility consists, in view of the results of Sect. 21.2.4, in imposing rank constraints in the moment matrices generated by the hierarchy. However, it is unclear how to do that efficiently, or how to relax such rank constraint in a way that preserves the SDP structure.

In this direction, it is important to mention that it is not guaranteed that a problem which is known to have a solution with finite dimension operators will generate a rank condition (21.22) in the hierarchy. That is, while from the solution involving finite dimension operators it is possible to construct moment matrices satisfying the rank condition, it could happen that these matrices are not the ones returned by the SDP solver. All these questions deserve further investigation.

The finite dimension issue is particularly relevant in the case of quantum correlations (the first example provided in the introduction). Consider the correlations obtained by two distant parties when each performs  $m$  measurements of  $r$  possible results on a bipartite quantum system. Can all correlations compatible with the quantum formalism be expressed using finite-dimensional operators? That is, do the algebraic properties satisfied by the operators in (21.1) impose any constraint on the relevant Hilbert space dimension? Some progress on this question has been realized in the last years. First, it was shown in [7] that  $r$ -dimensional operators are not enough to generate all quantum correlations with  $r$  possible results for an arbitrary number of measurements. This result was later generalized in [6, 45], where it was shown that Hilbert spaces of infinite dimension are needed when  $m \rightarrow \infty$  and  $r = 2$ . However, the question remains open when both  $m$  and  $r$  are finite.

Deciding about the possibility or impossibility to simulate general quantum correlations with finite dimensional resources is sometimes known in mathematical physics as Tsirelson's problem [54]. SDP hierarchies could play a role in its resolution. As mentioned in the introduction, Vértesi and Pál applied our SDP hierarchy to 241 Bell inequalities and showed that the hierarchy was providing the actual quantum value, up to numerical precision, for all but 16 inequalities. The most striking example was an inequality known as  $I_{3322}$  [13, 18, 55] which only involves three measurement of two outcomes,  $m = 3$  and  $r = 2$ . In spite of these small values for  $m$  and  $r$ , Vértesi and Pál observed a gap between the best known quantum value and the fourth SDP relaxation. Led by this result, they have more recently obtained a better quantum value by constructing a semi-analytic sequence of states and measurement operators of increasing dimension for which the  $I_{3322}$  value tends, up to computer precision, to the upper bound given by the fourth SDP relaxation [46]. These results give some support to the conjecture that infinite dimensional systems may be necessary to represent certain quantum correlations with  $m$  measurements of  $r$  results, even for finite  $m$  and  $r$ . More generally, instances where a gap is observed between the best known quantum solution and higher steps in our hierarchy are natural candidates to prove this conjecture.

### 21.3.2.1 Tensor Product Versus Commuting Operators

The above discussion is intimately connected to another open problem in mathematical physics. Consider the following variant of (21.1)

$$\begin{aligned} \sup_{E, \phi} \quad & \left\langle \phi, \sum_{ij} c_{ij} E_i \otimes E_j \phi \right\rangle \\ \text{subject to} \quad & E_i E_j = \delta_{ij} E_i \quad \text{for all } i, j \in M_k \text{ and for all } M_k, \\ & \sum_{i \in M_k} E_i = 1 \quad \text{for all } M_k, \end{aligned} \tag{21.58}$$

where  $E_i \in B(H_A)$ , for  $i \in A$ ;  $E_j \in B(H_B)$ , for  $j \in B$ ; and  $\phi \in H_A \otimes H_B$ . Since  $[E_i \otimes \mathbb{I}_B, \mathbb{I}_A \otimes E_j] = 0$ , problem (21.1) is a relaxation of (21.58), i.e.,  $I^c \geq I^t$ , where  $I^c$  denotes the optimal solution of (21.1) and  $I^t$  of (21.58). The composition of two systems in quantum theory is described by the tensor product of the Hilbert spaces of each system, and thus the tensor product formulation (21.58) is often preferred over the commuting formulation (21.1).

Tsirelson proved that both problems (21.1) and (21.58) are equivalent for the case of finite dimensional Hilbert spaces<sup>3</sup> [54]. On the other hand, it can be shown that, for any feasible solution  $(E, \phi)$  of problem (21.58), there exists a finite dimensional solution  $(\tilde{E}, \tilde{\phi})$  on  $\mathbb{C}^d$  with an objective value arbitrarily close to that of  $(E, \phi)$ , for  $d$  high enough. Tsirelson's problem is thus equivalent to check if the maximization problems (21.1) and (21.58) output the same values for any initial input  $\{c_{ij}\}$ .

Finding upper bounds on  $I^t$  for a given inequality  $\{c_{ij}\}$  could then lead to progress toward solving Tsirelson's problem. Unfortunately, it is not clear how to deal with tensor products in the SDP relaxations, other than relaxing them through commutativity constraints. In this context, an SDP hierarchy reformulation of problem (21.58) would be welcome. Given two sets of variables  $(x_1, y_2, \dots, x_n)$ ,  $(y_1, y_2, \dots, y_m)$ , consider the free  $*$ -algebra generated by  $(x_i \otimes y_j : i = 1, \dots, n; j = 1, \dots, m)$ . The problem at stake would be for which values of  $\lambda \in \mathbb{R}$  the polynomial  $p(X \otimes Y) - \lambda 1 \otimes 1$  is positive semidefinite for all operators  $(X_1, \dots, X_n, Y_1, \dots, Y_m)$  satisfying a given set of polynomial constraints  $\{q_i(X \otimes Y) \geq 0\}$ . It is not clear, though, that all positive polynomials in this scenario admit an SOS decomposition, so a new approach beyond moment and localizing matrices may be needed.

### 21.3.3 Unbounded Operators

The convergence properties that we have established here rely in an essential way on the Archimedean assumption, which guarantees that the operators defining the problem (21.13) are bounded. It would be interesting to develop methods to deal

---

<sup>3</sup>Actually, Pál and Vértesi's algorithms work by optimizing over the set of operators and states defined by (21.58).

with problems involving unbounded operators and for which the Archimedean assumption is not verified (taking, e.g., inspiration from unconstrained polynomial optimization [27]).

Consider, for instance, a one dimensional quantum particle of mass  $m$  moving along the  $q$  axis and subject to an arbitrary analytic potential  $V(q)$ . If  $V(q)$  has a global minimum at  $q_0$ , then classically one would expect the particle to oscillate around  $q_0$  in the low energy limit. Thus replacing  $V(q)$  by its truncated Taylor expansion around  $q_0$ , i.e.,  $V(q) \approx P(q) = \sum_{k=0}^{2N} \frac{V^{(k)}(q_0)}{k!}(q - q_0)^k$  should lead to similar physical predictions, provided that we choose  $N$  big enough. Under this approximation, the Hamiltonian operator governing the dynamics of the particle is

$$H(a, a^*) = -\frac{(a - a^*)^2}{4m} + P\left(\frac{a + a^*}{\sqrt{2}}\right), \quad (21.59)$$

where  $a$  is a non-hermitian operator such that  $[a, a^*] = \mathbb{I}$ . In these conditions, calculating the ground state energy  $E_0$  of this system can be cast as an NC polynomial optimization problem:

$$\begin{aligned} E_0 = \min_{X, \phi} \quad & \langle \phi, H(a, a^*) \phi \rangle \\ \text{subject to } & [a, a^*] = \mathbb{I}. \end{aligned} \quad (21.60)$$

In principle, this problem can be attacked using our SDP techniques. Due to the commutation properties of  $a, a^*$ , the correct Gröbner basis for this problem is the set of monomials  $\{(a^*)^m a^n : m, n \in \mathbb{N}\}$ . This is usually called normal ordering in physics and the corresponding polynomials are referred to as hereditary polynomials in mathematics (note that sums of squares for hereditary polynomials have been considered in [23]). Using normal ordering, the size of the moment matrix  $M_k(y)$  scales quadratically with  $k$ , and so it should be possible to implement high order relaxations of the problem (21.60).

However, in this case, the implementation of the hierarchy of SDPs faces two serious troubles: first of all, in the Schrödinger representation<sup>4</sup>,  $a$  is an unbounded operator. The Archimedean assumption does not hold, and therefore we cannot guarantee the convergence of the sequence of lower bounds on  $E_0$ . Secondly, any operator satisfying the commutation relation  $[a, a^*] = \mathbb{I}$  must necessarily act over an infinite dimensional Hilbert space  $H$ , so the optimality condition (21.22) will never be met<sup>5</sup>.

---

<sup>4</sup>In the Schrödinger representation for the operators  $a, a^*$ , the Hilbert space is  $\mathcal{L}^2(\mathbb{R})$ , and, for any  $f(x) \in \mathcal{L}^2(\mathbb{R})$ , the action of  $a$  over  $f(x)$  is given by  $af(x) = [xf(x) + f'(x)]/\sqrt{2}$ .

<sup>5</sup>Actually, it is not necessary to invoke unbounded operators to get situations similar to this second point: the commutation condition  $i[X, Y] \geq 0$  is satisfied in a non-trivial way only in infinite-dimensional spaces (see for instance [20]).

Since Theorems 21.1 and 21.2 in Sect. 2 do not apply to this case, other techniques have to be used in order to analyze the convergence of the SDP hierarchy. The performance and feasibility of this new numerical method to compute ground state energies, as well as its extensions to many-particle scenarios, will be the main topics of a future work [42]. See also the related works [12, 39, 52, 53].

### 21.3.4 Convergence Properties of Specific Problems

Many relevant optimization problems involve polynomials of small degree (for instance, Bell inequalities, particles described by two-body interactions, etc.) and particular types of operators (projectors, Pauli matrices, etc.). It is then an interesting question to focus on specific classes of problems and see what can be said about their convergence properties. Further, it would be interesting to understand if it is possible to tailor and improve the SDP hierarchy for such problems by exploiting their mathematical structure.

An example in this direction comes again from the problem of characterizing quantum correlations. Without entering into details, it is known that for problems (21.1) involving measurements of two outcomes, the first step of the hierarchy always yields the optimal value for a certain class of objective functions (corresponding to expressions that involve only “correlators”). This follows from previous results by Tsirelson [58] and was noted in [62]. For a more general class of problems (known as “unique games”), it has been shown in [29] that the first step of the hierarchy always provides a good approximation to the optimal value. Are there other families of objective functions where similar results hold?

Another example [49] involves the minimization of a polynomial evaluated over projection operators for which the first relaxation step always yields the optimal value. For simplicity focus on a polynomial  $p$  of degree 2 (but a similar argument holds for polynomials of any degree) and consider the quadratic problem

$$\begin{aligned} p^* = \min_{X, \phi} \quad & \langle \phi, p(X)\phi \rangle \\ \text{subject to} \quad & X_i^2 - X_i = 0 \quad i = 1, \dots, n, \end{aligned} \tag{21.61}$$

where the variables  $X_1, \dots, X_n$  are assumed to be hermitian. Its first order relaxation is

$$\begin{aligned} p^1 = \min_y \quad & \sum_w p_w y_w \\ \text{s.t.} \quad & y_1 = 1 \\ & M_1(y) \succeq 0 \\ & y_{ii} - y_i = 0 \quad i = 1, \dots, n. \end{aligned} \tag{21.62}$$

Any feasible point  $y$  of the above SDP problem defines a feasible point of (21.61) with objective value  $\langle \phi, p(X)\phi \rangle = \sum_w p_w y_w =: p(y)$ , and therefore  $p^1 = p^\star$ . To see this, perform a Gram decomposition of the matrix  $M_1(y)$ :  $M_1(y)(v, w) = y_{vw} = \langle \bar{v}, \bar{w} \rangle$ , where  $|v|, |w| \leq 1$ , i.e.,  $v, w \in \{1, x_1, \dots, x_n\}$ . Define the vector  $\phi = \bar{1}$ , which is normalized since  $\langle \bar{1}, \bar{1} \rangle = y_1 = 1$ , and the operators  $X_i$  ( $i = 1, \dots, n$ ) as the projectors on  $\bar{x}_i$ . Obviously,  $X_i^2 = X_i$ . Moreover,  $X_i \phi = X_i \bar{x}_i + X_i(\phi - \bar{x}_i) = \bar{x}_i$ , where the last equality follows from the fact that  $\langle \bar{x}_i, \phi - \bar{x}_i \rangle = y_i - y_{ii} = 0$ . This implies that  $y_{vw} = \langle \phi, v(X)w(X)\phi \rangle$  for  $v, w \in \{1, x_1, \dots, x_n\}$  and therefore that  $p(y) = \langle \phi, p(X)\phi \rangle$  since  $p$  is of degree 2.

Note that the properties of the problem can drastically modify the convergence of the hierarchy. This is dramatically illustrated by considering the commutative version of (21.61), which is the quadratically constrained quadratic program

$$\begin{aligned} p^\star &= \min_{x \in \mathbb{R}^n} && p(x) \\ \text{subject to} & && x_i^2 - x_i = 0 \quad i = 1, \dots, n. \end{aligned} \quad (21.63)$$

Since 0-1 integer programming can be formulated in this form, it is NP-hard to solve a general instance of (21.63). Thus, contrary to the non-commutative case, it is highly unlikely that considering relaxations up to some bounded order might be sufficient to solve this problem.

### 21.3.5 Mutually Unbiased Bases

The optimization method presented here can be adapted to feasibility problems where the aim is to decide if there exists a set of operators satisfying a set of algebraic constraints. More precisely, the method can be used to certify that no operators satisfying the algebraic constraints exist: this is the case when one of the corresponding SDP relaxations has no solution (up to numerical precision).

One problem of this form to which our hierarchy of relaxations could be applied is the question of whether there exist  $d+1$  mutually unbiased bases (MUB) in a complex Hilbert space of finite dimension  $d$  [16]. Consider two orthonormal bases in this space,  $\{e_i\}$  and  $\{f_j\}$ , with  $i = 1, \dots, d$ . These bases are said to be mutually unbiased whenever

$$|\langle e_i, f_j \rangle|^2 = \frac{1}{d}, \quad \text{for all } i, j. \quad (21.64)$$

MUB's encapsulate the concept of complementarity in quantum theory. Recall that when performing the measurement described by a basis  $\{e_i\}$  on a system in state  $\psi$ , there are  $d$  possible results, corresponding to each element of the basis, and each result occurs with probability  $p(i) = |\langle e_i, \psi \rangle|^2$ . Thus, if an element of a basis is measured along another basis which is mutually unbiased, the  $d$  possible results occur with the same probability, namely  $1/d$ .

Beyond their fundamental interest, MUB's find applications in several problems in quantum information theory. First of all, measuring along these bases provides information about an unknown quantum state in a non-redundant way. That is, knowing the probabilities for the elements in one basis does not allow to recover in general the probabilities for a measurement in another mutually unbiased basis. This allows achieving the best quantum estimation results, as the measurement in one basis duplicate no information already contained in other MUB measurements. They also play an important role in quantum cryptography, as the best known protocols in systems of arbitrary dimension use these bases [11]. Finally, from a more fundamental viewpoint, they also provide the solution to the “Mean King’s” problem [17].

It is known that in systems having a dimension equal to a power of prime number there are  $d + 1$  MUB's [26, 63]. Actually, this is an upper bound to the number of such bases for any dimension. Since the beginning of the 80's, the open problem is to determine whether this bound is attainable for arbitrary dimension. The first interesting case is  $d = 6$ , as smaller dimensions correspond to powers of prime numbers. The question, then, is whether there exist seven MUB's in a six-dimensional complex Hilbert space. Despite years of efforts, there has been little improvement on this problem [16]. However, recent works give numerical evidence for the existence of only 3 MUB's in systems of dimension six [5, 9]. The numerical evidence is far from being conclusive as (1) it consists of a numerical optimization over a big space depending of many parameters, and (2) the structure of the optimization does not avoid the possibility of local minima.

The hierarchy of SDP conditions introduced here can be applied to solve the MUB's problem. In what follows, we consider  $K$  bases which are mutually unbiased and denote the vectors by  $e_i^x$ , where  $x = 1, \dots, K$  labels the basis and  $i = 1, \dots, d$ , the vector element within the basis  $x$ . Because of condition (21.64), the projectors onto these vectors,  $\Pi_i^b = e_i^b(e_i^b)^*$  satisfy the following relations:

1.  $\Pi_i^x = (\Pi_i^x)^*$ ;
2.  $\Pi_i^x \Pi_i^{x'} = \delta_{ij} \Pi_i^x$ ;
3.  $\sum_{i=1}^d \Pi_i^x = \mathbb{I}$ ;
4.  $\Pi_i^x \Pi_j^{x'} \Pi_i^x = \frac{1}{d} \Pi_i^x$ , for  $x \neq x'$ ;
5.  $[\Pi_i^x O_1 \Pi_i^x, \Pi_i^x O_2 \Pi_i^x] = 0$ , for any  $x, i$  and any pair of monomials  $O_1$  and  $O_2$  of  $\{\Pi_i^x\}$  of length three.<sup>6</sup>

Clearly, the existence of  $K$  MUB's in a system of dimension  $d$  implies the existence of operators  $\Pi_i^x$  satisfying the previous relations. The following theorem shows that the converse is also true: it is possible to derive from operators  $\Pi_i^x$  compatible with conditions 1–5,  $K$  MUB's in dimension  $d$ .

**Theorem 21.3.** *Let  $\{\Pi_i^x\}$  be a set of operators acting over a separable Hilbert space  $H$  satisfying conditions 1–5. Then, there exists a partition of the Hilbert*

---

<sup>6</sup>In fact, this condition holds for any pair of arbitrary operators  $O_1$  and  $O_2$ .

space  $H = \int^{\oplus} d\lambda H_{\lambda}$ , with  $H_{\lambda} \cong \mathbb{C}^d$ , and a unitary transformation  $W$  such that  $W\Pi_i^x W^* = \int^{\oplus} d\lambda \Pi_{i,\lambda}^x$ , where each  $\Pi_{i,\lambda}^x$  is a rank-1 projector.

The proof of this Theorem is given at the end of this subsection. It establishes the equivalence between the original problem of the existence of  $K$  MUB's in a space of given dimension and the existence of a set of operators satisfying conditions 1–5. This second formulation, however, can be tackled using our hierarchy of conditions because no constraint is imposed on the dimension of the space on which the operators are defined. Unfortunately, at the moment, the number of parameters needed in the first non-trivial case,  $d = 6$ , is still too large for standard computational capabilities.

Before concluding this section, we would like to mention that a different approach to the MUB problem using Lasserre's optimization techniques has been proposed in [4]. This approach has similar scaling problems as ours and the case of  $d = 6$  also becomes intractable using state-of-the-art computers. That is, the two approaches could potentially solve the MUB problem if better computational capabilities were available. We hope however that our approach may be more advantageous from a computational point of view, as it captures the non-commutativity nature of the original problem.

### 21.3.5.1 Proof of Theorem 21.3

*Proof.* Let  $\{\Pi_i^x\}$  be a set of operators satisfying conditions 1–5. Since the projectors  $\{\Pi_i^1 : i = 1, \dots, d\}$  commute, they can be diagonalized simultaneously. The total space  $\mathcal{H}$  can then be written as  $\mathcal{H} = \bigoplus_{i=1}^d \mathcal{H}^i$ , where each  $\Pi_k^1$  is such that  $\Pi_k^1 = (\bigoplus_{i=1}^{k-1} 0^i) \oplus \mathbb{I}^k \oplus (\bigoplus_{i=k+1}^d 0^i)$ . We prove in what follows that all  $\mathcal{H}^k$  are isomorphic.

By virtue of condition 3 we can decompose any projector  $\Pi_j^y$  with  $y \neq 1$  as

$$\Pi_j^y = \sum_{j,k=1}^d \Pi_j^1 \Pi_i^y \Pi_k^1 = \sum_{j,k=1}^d \frac{1}{d} \tilde{O}_{j,k}^{y,i}, \quad (21.65)$$

where each  $\tilde{O}_{j,k}^{y,i}$  is an operator  $\mathcal{H}^k \rightarrow \mathcal{H}^j$ . Note that, from  $\Pi_i^y = (\Pi_i^y)^*$ , it follows that  $\tilde{O}_{j,k}^{y,i} = (\tilde{O}_{k,j}^{y,i})^*$ . Also, we have that

$$\tilde{O}_{j,k}^{y,i} (\tilde{O}_{j,k}^{y,i})^* = d^2 \cdot \Pi_j^1 \Pi_i^y \Pi_k^1 \Pi_k^1 \Pi_i^y \Pi_j^1 = \Pi_j^1, \quad (21.66)$$

where the last equality follows from invoking conditions 2 and 4. Analogously,  $(\tilde{O}_{j,k}^{y,i})^* \tilde{O}_{j,k}^{y,i} = \Pi_k^1$ . The operator  $\tilde{O}_{j,k}^{y,i}$  is thus unitary and so the spaces  $\{\mathcal{H}^k\}$  are isomorphic to an (in general) infinite dimensional Hilbert space  $\tilde{\mathcal{H}}$  i.e.,  $\mathcal{H} = \bigoplus_{i=1}^d \tilde{\mathcal{H}}$ . Putting all these facts together, we have that operators  $\Pi_i^x$  can be expressed as

$$\Pi_1^1 = \begin{pmatrix} \mathbb{I} & & \\ 0 & \ddots & \\ & & 0 \end{pmatrix}, \Pi_2^1 = \begin{pmatrix} 0 & & \\ \mathbb{I} & \ddots & \\ & & 0 \end{pmatrix}, \dots, \Pi_d^1 = \begin{pmatrix} 0 & 0 & \\ & \ddots & \\ & & \mathbb{I} \end{pmatrix}, \quad (21.67)$$

while, for  $y \neq 1$ ,

$$\Pi_j^y = \frac{1}{d} \begin{pmatrix} \mathbb{I} & O_{12}^{y,j} & O_{13}^{y,j} & & \\ O_{21}^{y,j} & \mathbb{I} & O_{23}^{y,j} & & \\ O_{31}^{y,j} & O_{32}^{y,j} & \mathbb{I} & & \\ & & & \ddots & \\ & & & & \ddots \end{pmatrix}, \quad (21.68)$$

where the operators  $O_{j,k}^{y,i} : \tilde{\mathcal{H}} \rightarrow \tilde{\mathcal{H}}$  correspond to the non-trivial part of  $\tilde{O}_{j,k}^{y,i}$ .

Now, condition 4 implies that

$$O_{kl}^{y,j} = \frac{1}{d} O_{k1}^{y,j} O_{l1}^{y,j}. \quad (21.69)$$

This last equation states that all operators  $\{O_{kl}^{y,j}\}$  can be derived from  $\{O_{k1}^{y,j}\}$ . Perform a change of basis according to the unitary  $U = \bigoplus_{i=1}^d O_{1,i}^{2,1}$ ,  $\Pi_j^y \rightarrow U \Pi_j^y U^*$ . In this new basis, the operators  $\Pi_j^x$  have still the form (21.67), (21.68), but  $\Pi_1^2$  equals

$$\Pi_1^2 = \frac{1}{d} \begin{pmatrix} \mathbb{I} & \mathbb{I} & \mathbb{I} & & \\ \mathbb{I} & \mathbb{I} & \mathbb{I} & & \\ \mathbb{I} & \mathbb{I} & \mathbb{I} & & \\ & & & \ddots & \\ & & & & \ddots \end{pmatrix}. \quad (21.70)$$

Defining  $A_{ij} \equiv \Pi_i^1 \Pi_1^2 \Pi_j^1$  one can check that

$$A_{1j} \Pi_i^y A_{k1} = \frac{1}{d^3} \begin{pmatrix} O_{jk}^{y,i} & 0 & 0 & & \\ 0 & 0 & 0 & & \\ 0 & 0 & 0 & & \\ & & & \ddots & \\ & & & & \ddots \end{pmatrix}. \quad (21.71)$$

Due to Condition 5, the commutator  $[A_{1j} P^{y,i} A_{k1}, A_{1m} P^{z,l} A_{n1}]$  can be shown to be zero. Indeed,

$$\begin{aligned} A_{1j} \Pi_i^y A_{k1} A_{1m} \Pi_l^z A_{n1} &= \Pi_1^1 \Pi_1^2 \Pi_j^1 \Pi_i^y \Pi_k^1 \cdot (\Pi_1^2 \Pi_1^1 \Pi_1^2) \cdot \Pi_m^1 \Pi_l^z \Pi_n^1 \Pi_1^2 \Pi_1^1 \\ &= \frac{1}{d} \Pi_1^1 \Pi_1^2 \Pi_j^1 \Pi_i^y \Pi_k^1 \Pi_1^2 \Pi_m^1 \Pi_l^z \Pi_n^1 \Pi_1^1 \Pi_1^2 \Pi_1^1 \\ &= \frac{1}{d} \Pi_1^1 (\Pi_1^2 \Pi_j^1 \Pi_i^y \Pi_k^1 \Pi_1^2) (\Pi_1^2 \Pi_m^1 \Pi_l^z \Pi_n^1 \Pi_1^2) \Pi_1^1 \end{aligned}$$

$$\begin{aligned}
&= \frac{1}{d} \Pi_1^1 (\Pi_1^2 \Pi_m^1 \Pi_l^z \Pi_n^1 \Pi_1^2) (\Pi_1^2 \Pi_j^1 \Pi_i^y \Pi_k^1 \Pi_1^2) \Pi_1^1 \\
&= A_{1m} \Pi_l^z A_{n1} \cdot A_{1j} \Pi_i^y A_{k1},
\end{aligned} \tag{21.72}$$

where Conditions 2, 4 and 5 were respectively used in the first and third, second and fourth equality. In the light of (21.71), the previous expression implies that

$$[O_{jk}^{y,i}, O_{m,n}^{z,l}] = 0, \text{ for all } y,z,i,j,k,l,m,n, \tag{21.73}$$

and, therefore, there exists a basis where all  $\{O_{jk}^{y,i}\}$  can be simultaneously diagonalized. Let  $\tilde{V} \in B(\tilde{\mathcal{H}})$  be the unitary operation that implements such a change of basis. Then, the unitary  $V = \bigoplus_{i=1}^d \tilde{V} \in B(\mathcal{H})$  is such that the redefined projectors  $\Pi_i^y \rightarrow V \Pi_i^y V^*$  have the form

$$\begin{aligned}
\Pi_1^1 &= \begin{pmatrix} \int d\lambda \lambda(\lambda)^* & & \\ & 0 & \\ & & 0 \\ & & \ddots \end{pmatrix}, \quad \Pi_2^1 = \begin{pmatrix} 0 & \int d\lambda \lambda(\lambda)^* & & \\ & 0 & & \\ & & 0 & \\ & & & \ddots \end{pmatrix}, \dots, \\
\Pi_j^y &= \frac{1}{d} \begin{pmatrix} \int d\lambda \lambda(\lambda)^* & D_{12}^{y,j} & D_{13}^{y,j} & & \\ D_{21}^{y,j} & \int d\lambda \lambda(\lambda)^* & D_{23}^{y,j} & & \\ D_{31}^{y,j} & D_{32}^{y,j} & \int d\lambda \lambda(\lambda)^* & & \\ & & & 0 & \\ & & & & \ddots \end{pmatrix}, \text{ for } y \neq 1,
\end{aligned} \tag{21.74}$$

where  $\{\lambda\}$  is an orthonormal basis for  $\tilde{\mathcal{H}}$  and  $D_{kl}^{y,j} = \int d\lambda e^{i\theta_{kl}^{y,j}(\lambda)} \lambda(\lambda)^*$ . Fix  $\lambda$ , and consider the matrices

$$\begin{aligned}
\Pi_{1,\lambda}^1 &= \begin{pmatrix} 1 & & & \\ 0 & & & \\ & 0 & & \\ & & \ddots & \end{pmatrix}, \quad \Pi_{2,\lambda}^1 = \begin{pmatrix} 0 & & & \\ & 1 & & \\ & & 0 & \\ & & & \ddots \end{pmatrix}, \dots, \\
\Pi_{j,\lambda}^y &= \frac{1}{d} \begin{pmatrix} 1 & e^{i\theta_{12}^{y,j}(\lambda)} & e^{i\theta_{13}^{y,j}(\lambda)} & & \\ e^{i\theta_{21}^{y,j}(\lambda)} & 1 & e^{i\theta_{23}^{y,j}(\lambda)} & & \\ e^{i\theta_{31}^{y,j}(\lambda)} & e^{i\theta_{32}^{y,j}(\lambda)} & 1 & & \\ & & & 0 & \\ & & & & \ddots \end{pmatrix}, \text{ for } y \neq 1.
\end{aligned} \tag{21.75}$$

Because of the orthogonality relations between the vectors  $\lambda$ , conditions 1–5 are also satisfied by these  $d \times d$  matrices. Clearly, the matrices  $\Pi_{i,\lambda}^1$  are 1-rank projectors, and the fact that  $\Pi_{j,\lambda}^y \Pi_{1,\lambda}^1 \Pi_{j,\lambda}^y = \Pi_{j,\lambda}^y / d$ , for  $y \neq 1$  implies that so are the rest of projectors  $\Pi_{j,\lambda}^y$ 's. These projectors define  $K$  MUB's in a  $d$ -dimensional system.

### 21.3.6 Large-Scale SDP Solvers

Finally, we point out that while the approach on SDP relaxations presented here has already proven useful for different type of problems (mainly in the context of quantum correlations as discussed in the introduction), the main factor that limits its wider applicability (particularly for quantum chemistry and more generally many-body physics) is the computational time and memory resources needed to solve large SDP programs. From this perspective, it would be extremely valuable to develop more efficient large-scale SDP solvers.

## References

1. Allcock, J., Brunner, N., Pawłowski, M., Scarani, V.: Recovering part of the boundary between quantum and nonquantum correlations from information causality. *Phys. Rev. A* **80**, 040103 (2009)
2. Almeida, M.L., Bancal, J.-D., Brunner, N., Acín, A., Gisin, N., Pironio, S.: Guess your neighbors input: A multipartite nonlocal game with no quantum advantage. *Phys. Rev. Lett.* **104**, 230404 (2010)
3. Bell, J.S.: Speakable and unspeakable in quantum mechanics. Cambridge University Press, Cambridge (1987)
4. Briërley, S., Weigert, S.: Mutually unbiased bases and semi-definite programming. arXiv:1006.0093.
5. Briërley, S., Weigert, S.: Maximal sets of mutually unbiased quantum states in dimension 6. *Phys. Rev. A* **78**, 042312 (2008)
6. Briet, J., Buhrman, H., Toner, B.: A generalized grothendieck inequality and entanglement in xor games. arXiv:0901.2009.
7. Brunner, N., Pironio, S., Acín, A., Gisin, N., Méhot, A.A., Scarani, V.: Testing the dimension of hilbert spaces. *Phys. Rev. Lett.* **100**, 210503 (2008)
8. Buhrman, H., Cleve, R., Wolf, R.: Nonlocality and communication complexity. *Rev. Mod. Phys.* **82**, 665–698 (2010)
9. Butterley, P., Hall, W.: Numerical evidence for the maximum number of mutually unbiased bases in dimension six. arXiv:quant-ph/0701122.
10. Cafuta, K., Klep, I., Povh, J.: NCSOStools: a computer algebra system for symbolic and numerical computation with noncommutative polynomials. <http://ncsostools.fis.unm.si>.
11. Cerf, N., Bourennane, M., Karlsson, A., Gisin, N.: Security of quantum key distribution using d-level systems. *Phys. Rev. Lett.* **88**, 127902 (2005)
12. Cimprič, J.: A method for computing lowest eigenvalues of symmetric polynomial differential operators by semidefinite programming. *J. Math. Anal. Appl.* **369**, 443–452 (2010)
13. Collins D., Gisin, N.: A relevant two qubit bell inequality inequivalent to the chsh inequality. *J. Phys. A: Math. Gen.* **37**, 1775–1787 (2004)
14. Curto, R.E., Fialkow, L.A.: Solution of the truncated complex moment problem with flat data. *Mem. Amer. Math. Soc.* **119**, 568–619 (1996)
15. Doherty, A.C., Liang, Y.C., Toner, B., Wehner, S.: The quantum moment problem and bounds on entangled multi-prover games. In: Proceedings of IEEE Conference on Computational Complexity, College Park, Maryland, 23-26 June 2008
16. Durt, T., Englert, B.-G., Bengtsson, I., Zyczkowski, K.: On mutually unbiased bases. arXiv:1004.3348.

17. Englert, B.G.Y., Aharonov, Y.: The mean king's problem: Spin 1. *Zeitschrift fur Naturforschung* **56a**, 16–19 (2001)
18. Fölling, M.: Constructive generalization of bell's inequalities. *Nuov. Cim. B* **64**, 241–251 (1981)
19. Fukuda, M., Braams, B.J., Nakata, M., Overton, M.L., Percus, J. K., Yamashita, M., Zhao, Z.: Large-scale semidefinite programs in electronic structure calculation. *Math. Program. Ser. B* **109**, 553–580 (2007)
20. Halmos, P.R.: A hilbert space problem book. Springer-Verlag, New York (1982)
21. Helton, J.W.: “positive” noncommutative polynomials are sums of squares. *Ann. of Math. 2nd Ser.* **156**, 675–694 (2002)
22. Helton, J.W., McCullough, S.A.: A positivstellensatz for non-commutative polynomials. *Trans. Amer. Math. Soc.* **356**, 3721–3737 (2004)
23. Helton, J.W., McCullough, S.A., Putinar, M.: Non-negative hereditary polynomials in a free \*-algebra. *Math. Z.* **250**, 515–522 (2005)
24. Henrion D., Lasserre, J.B.: Detecting global optimality and extracting solutions in gloptipoly. In: Henrion, D., Garulli, A. (eds.) *Positive Polynomials in Control*, vol. 149 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin-Heidelberg (2005)
25. Horn, R.A., Johnson, C.R.: *Matrix Analysis*. Cambridge University Press, Cambridge (1999)
26. Ivanovic, I.D.: Geometrical description of quantal state determination. *J. Phys. A: Math. Gen.* **14**, 3241–3245 (1981)
27. Jibetean, D., Laurent, M., Semidefinite approximations for global unconstrained polynomial optimization. *SIAM J. Optim.* **16**(2), 490–514 (2005)
28. Junge, M., Palazuelos, C., Pérez-García, D., Villanueva, I., Wolf, M.M.: Operator space theory: A natural framework for bell inequalities. *15*, 170405 (2010)
29. Kempe, J., Regev, O., Toner, B.: Unique games with entangled provers are easy. In: *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, 457–466 (2008)
30. Klep, I., Schweighofer, M.: A nichtnegativstellensatz for polynomials in noncommuting variables. *Israel J. Math.* **161**(1), 17–27 (2007)
31. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. *SIAM J. Optim.* **11**, 796–817 (2001)
32. Laurent, M.: Sums of squares, moment matrices and optimization over polynomials. In: Putinar, M., Sullivant, S. (eds.) *Emerging Applications of Algebraic Geometry*, vol. 149 of *IMA Volumes in Mathematics and its Applications*. Springer-Verlag, Berlin-Heidelberg (2009)
33. Liang, Y.-C., Lim, C.-W., Deng, D.-L.: Reexamination of a multisetting bell inequality for qudits. *Phys. Rev. A* **80**, 052116 (2009)
34. Löfberg, J.: *YALMIP* : A Toolbox for Modeling and Optimization in MATLAB. <http://control.ee.ethz.ch/~joloef/yalmip.php>.
35. Masanes, L., S. Pironio, S., Acín, A.: Secure device-independent quantum key distribution with causally independent measurement devices. *Nature Comms.* **2**, 184 (2011); also available at arxiv:1009.1567
36. Mazzotti, D.A.: Realization of quantum chemistry without wave functions through first-order semidefinite programming. *Phys. Rev. Lett.* **93**, 213001 (2004)
37. Mazzotti, D.A. (ed.): *Reduced-Density-Matrix Mechanics: With Application to Many-Electron Atoms and Molecules*, vol. 134 of *Advances in Chemical Physics*. Wiley, New York (2007)
38. Mora, T.: An introduction to commutative and noncommutative gröbner bases. *Theor. Comput. Sci.* **134**, 131–173 (1994)
39. Nahas, J.: On the positivstellensatz in weyl's algebra. *Proc. Amer. Math. Soc.* **138**, 987–995 (2010)
40. Navascués, M., Pironio, S., Acín, A.: Bounding the set of quantum correlations. *Phys. Rev. Lett.* **98**, 010401 (2007)

41. Navascués, M., Pironio, S., Acín, A.: A convergent hierarchy of semidefinite programs characterizing the set of quantum correlations. *New J. Phys.* **10**, 073013 (2008)
42. Navascués, M., Plenio, M.B., García-Sáez, A., Pironio, S., Acín, A.: Article in preparation.
43. Navascués, M., Wunderlich, H.: A glance beyond the quantum model. *Proc. Roy. Soc. Lond. A* **466**, 881–890, (2009)
44. Pál, K.F., Vértesi, T.: Quantum bounds on bell inequalities. *Phys. Rev. A* **79**, 022120 (2008)
45. Pál, K.F., Vértesi, T.: Bounding the dimension of bipartite quantum systems. *Phys. Rev. A* **79**, 042106 (2009)
46. Pál, K.F., Vértesi, T.: Maximal violation of a bipartite three-setting, two-outcome bell inequality using infinite-dimensional quantum systems. *Phys. Rev. A* **82**, 022116 (2010)
47. Parrilo, P.A.: Semidefinite programming relaxations for semialgebraic problems. *Math. Program. Ser. B* **96**, 293–320 (2003)
48. Pironio, S., Acín, A., Massar, S., Boyer de la Giroday, A., Matsukevich, D.N., Maunz, P., Olmschenk, S., Hayes, D., Luo, L., Manning, T.A., Monroe, C.: Random numbers certified by bells theorem. *Nature* **464**, 1021–1024 (2010)
49. Pironio, S., Navascués, M., Acín, A.: Convergent relaxations of polynomial optimization problems with noncommuting variables. *SIAM J. Optim.* **20**, 2157–2180 (2010)
50. Pisier, G.: Introduction to Operator Space Theory. Cambridge Univ. Press, Cambridge (2003)
51. Putinar, M.: Positive polynomials on compact semi-algebraic sets. *Indiana Univ. Math. J.* **42**, 969–984 (1993)
52. Savchuk, Y., Schmüdgen, K.: On unbounded induced representations of  $*$ -algebras. arXiv:0806.2428v1.
53. Schmüdgen, K.A.: Strict positivstellensatz for the weyl algebra. *Math. Ann.* **331**, 779–794 (2005)
54. Scholz, V.B., Werner, R.F.: Tsirelson’s problem. arXiv:0812.4305.
55. Śliwa, C.: Symmetries of the bell correlation inequalities. *Phys. Lett. A* **317**, 165–168 (2003)
56. Sturm, J.F.: SeDuMi, a MATLAB toolbox for optimization over symmetric cones. <http://sedumi.mcmaster.ca>.
57. Szabo, A., Ostlund, N.S.: Modern Quantum Chemistry: introduction to advanced electronic structure theory. Dover publications Inc., Mineola, New York (1996)
58. Tsirelson, B.S.: Some results and problems on quantum bell-type inequalities. *Hadronic J. Supp.* **8**(4),329–345 (1993)
59. Vértesi, T., Pironio, S., Brunner, N.: Closing the detection loophole in bell experiments using qudits. *Phys. Rev. Lett.* **104**, 060401 (2010)
60. Voiculescu, D.: Free probability theory. American Mathematical Society, Providence, Rhode Island, USA (1997)
61. von Neumann, J.: On rings of operators. reduction theory. *Ann. of Math. 2nd Series* **50**, 401–485 (1949)
62. Wehner, S.: Tsirelson bounds for generalized clauser-horne-shimony-holt inequalities. *Phys. Rev. A* **73**, 022110 (2006)
63. Wootters, W.K., Fields, B.D.: Optimal state-determination by mutually unbiased measurements. *Ann. Phys.* **191**, 363–381 (1989)

# Chapter 22

## Semidefinite Programming and Constraint Programming

Willem-Jan van Hoeve

### 22.1 Introduction

Constraint programming is a modeling and solving paradigm for combinatorial optimization problems. In constraint programming, a combinatorial problem is modeled using variables that have an associated domain of possible values, and constraints over these variables. What makes constraint programming different from similar methods such as integer programming, is that the variable domains can be any finite set of elements, and the constraints can be of any type. For example, a constraint can be defined explicitly by a list of allowed tuples, or implicitly by an expression such as `alldifferent( $x_1, x_2, \dots, x_n$ )`, which specifies that the variables  $x_1, x_2, \dots, x_n$  take pairwise different values. In contrast, integer linear programming restricts the variables to take either a continuous or integer valued domain (in fact, an interval), while constraints can be only linear expressions. Hence, constraint programming offers a very rich, and often convenient, modeling language.

In order to solve a given model, a constraint programming solver implicitly enumerates all possible variable-value combinations through a systematic search process until a solution satisfying all constraints has been found (or until it is proved that no solution exists). Most constraint programming systems allow the user to specify a particular search strategy. For example, criteria can be expressed to determine the variable order to branch on, as well as the value order to assign to those variables.

---

W.-J. van Hoeve (✉)

Tepper School of Business, Carnegie Mellon University, 5000 Forbes Avenue,  
Pittsburgh, PA, USA

e-mail: [vanhoeve@andrew.cmu.edu](mailto:vanhoeve@andrew.cmu.edu)

To reduce the underlying exponentially large search space, so-called *domain filtering* algorithms are applied during the systematic search process. Each constraint has an associated filtering algorithm. Its task is to remove provably inconsistent values from the domains of the variables that are in the scope of the constraint, based on the individual constraint only. When a filtering algorithm for one constraint has processed the variables in its scope, the updated domains are propagated to the other constraints that share these variables, whose filtering algorithms are in turn activated. This cascading effect is called constraint propagation. Whenever a filtering algorithm makes a domain empty, we know that at that point of the search no solution is possible, and we backtrack to a previous search state.

By design, the inference of domain filtering is based on feasibility of each domain value. For optimization problems, similar inference can be performed by means of *cost-based* domain filtering. That is, a domain value is considered infeasible if assigning it to its variable will always yield a suboptimal solution (if any). Focacci et al. [13] introduced a systematic way of designing cost-based domain filtering algorithms for so-called ‘optimization constraints’, through embedding an associated relaxation. Although they apply linear relaxations, conceptually any suitable relaxation can be embedded, including semidefinite relaxations, as proposed by Van Hoeve [65, 66].

From a constraint programming perspective, there are two major reasons why semidefinite relaxations could be successfully integrated. First, the bounds obtained can be much tighter than those obtained with linear relaxations, and can help to improve the optimization reasoning. Second, the fractional solution of the semidefinite relaxation can be applied as an accurate search heuristic. It should be noted that the intention here is not to solve a semidefinite relaxation at every node of the search tree necessarily. Instead, a semidefinite relaxation may be solved only once at the root node, or at selected times during the search process.

Also from a semidefinite programming perspective, the integration with constraint programming can be beneficial. Namely, it is not uncommon that solving a semidefinite relaxation takes so much time that embedding it inside a pure SDP-based branch-and-bound scheme does not pay off. Instead, the approaches studied in this chapter focus on a limited number of relaxations to be solved, while different information is extracted and utilized from the solution, namely for domain filtering and search heuristics. These ideas may be applied also inside a pure SDP-based solver.

The remainder of the chapter is structured as follows. In Sect. 22.2 we present necessary background information on constraint programming. In Sect. 22.3 we describe how semidefinite relaxations have been applied within constraint programming. Section 22.4 presents a first application of this, for stable set problems. In Sect. 22.5 we discuss the accuracy of semidefinite relaxations when applied as a search heuristic. In Sect. 22.6, an integrated approach to a telecommunications application is presented, corresponding to the biclique completion problem. Finally, Sect. 22.7 presents conclusions and future research perspectives.

## 22.2 Constraint Programming

We first introduce basic constraint programming concepts that are necessary for this chapter. For more information on constraint programming we refer to the textbooks by Apt [4] and Dechter [8], and to the Handbook on Constraint Programming by Rossi et al. [60]. For more information on the integration of constraint programming and operations research we refer to Hooker [33, 35] and Milano [48].

### 22.2.1 Modeling

Even though constraint programming has many similarities with integer linear programming (ILP), there are some basic differences, also in terminology, that may cause confusion.

First, ILP assumes that variables take a value from a continuous interval represented by a lower and upper bound. In ILP systems, a variable is considered to be integer if its value is within a small tolerance factor of an integer number. In contrast, a CP variable takes its value from a specific finite set called its *domain*. For a variable  $x$ , we denote its domain by  $D(x)$ . Variable domains are represented explicitly in CP systems. This is a fundamental difference between ILP systems and CP systems. Namely, the explicit domain representation of CP not only allows more expressive models, but also allows to represent high-level inferences (e.g., we can remove individual values from a variable domain). On the other hand, the continuity assumption and linearity of ILP models allow the use of efficient linear programming solvers, but inferences are limited to those that can be represented by linear constraints and changes in the bounds of variables.

In CP, constraints can take several forms. Formally, a constraint  $C$  on a set of variables  $X = \{x_1, x_2, \dots, x_k\}$  is defined as a subset of the Cartesian product of the domains of the variables in  $X$ , i.e.,  $C \subseteq D(x_1) \times D(x_2) \times \dots \times D(x_k)$ . A tuple  $(d_1, \dots, d_k) \in C$  is called a *solution* to  $C$ . An immediate form to represent constraints is thus to list out the individual tuples that form the definition, and we say that such constraints are given ‘in extension’. For several applications, such as database applications, such representation is very natural and useful. However, often we prefer to model combinatorial problems using implicit formulations, for example using linear expressions. Most constraint programming systems allow constraints to be defined by any algebraic expression, using common operators such as addition and multiplication, but also logical connectives in which sub-expressions serve as Boolean functions, whose truth value can be interpreted as a binary value. For example, the constraint

$$z = \sum_{i=1}^n (x_i \geq b)$$

restricts variable  $z$  to be equal to the number of variables  $x_1, \dots, x_n$  that take a value of at least  $b$ . Furthermore, variables can appear as subscripts. For example, for an index set  $I$ , consider an array  $c \in \mathbb{Q}^{|I|}$ , a variable  $x$  with domain  $D(x) = I$ , and a variable  $z$ . The expression  $z = c_x$  states that  $z$  will be assigned the  $x$ -th value in the array  $c$ . Such constraints, in which a variable functions as an index, are called ‘element’ constraints.

Constraint programming further allows the use of so-called *global constraints* [58, 67]. Global constraints are defined on an arbitrary number of variables, and usually encapsulate a combinatorial structure that can be exploited during the solving process. A well-known example is the constraint `alldifferent`( $x_1, x_2, \dots, x_n$ ), that requires the variables  $x_1, x_2, \dots, x_n$  to take pairwise different values. Global constraints and element constraints can lead to very concise formulations. For example, consider a Traveling Salesman Problem on  $n$  cities, in which the distance between two cities  $i$  and  $j$  is denoted by  $d_{i,j}$ . The problem asks to find a closed tour visiting each city exactly once, with minimum total distance. We introduce a variable  $x_i$  representing the  $i$ -th city to be visited, for  $i = 1, \dots, n$ . The problem can then be formulated as

$$\begin{aligned} & \min \sum_{i=1}^{n-1} d_{x_i, x_{i+1}} + d_{x_n, x_1} \\ & \text{s.t. } \text{alldifferent}(x_1, x_2, \dots, x_n), \\ & \quad D(x_i) = \{1, 2, \dots, n\}, \quad \forall i \in \{1, \dots, n\}. \end{aligned}$$

In addition to global constraints, most CP systems contain other high-level modeling objects. For example, *set variables* are variables that will be assigned a set of elements [21, 51]. The domain of a set variable is a finite set of sets, that however often is exponentially large. Therefore, CP systems usually represent the domain of a set variable by a ‘lower bound’ of mandatory elements and an ‘upper bound’ of possible elements. In addition, a lower and upper bound on the cardinality of the set is maintained as part of the domain information. For example, if the domain of a set variable  $S$  is defined as  $D(S) = [\{a,c\}, \{a,b,c,d,e\}]$  with  $3 \leq |S| \leq 4$ , the mandatory elements are  $\{a,c\}$  while the possible elements are  $\{a,b,c,d,e\}$  and in addition the cardinality must be between 3 and 4. That is, the domain implicitly includes the sets  $\{a,b,c\}, \{a,c,d\}, \{a,c,e\}, \{a,b,c,d\}, \{a,b,c,e\}$ , and  $\{a,c,d,e\}$ . As an illustration, we next model the weighted stable set problem using a set variable. Let  $G = (V, E)$  be an undirected weighted graph with vertex set  $V$ , edge set  $E$ , and a ‘weight’ function  $w : V \rightarrow \mathbb{R}$ . The maximum weighted stable set problem asks to find a subset of vertices with maximum total weight, such that no two vertices in this subset share an edge. We introduce one set variable  $S$  representing the vertices that will form the stable set. The initial domain of this set variable is  $D(S) = [\emptyset, V]$ , and  $0 \leq |S| \leq |V|$ . That is, initially there are no mandatory elements, and all elements

in  $V$  can potentially be in the stable set. The constraint programming model then becomes

$$\begin{aligned} \max \quad & \sum_{i \in S} w_i \\ \text{s.t. } & (i \in S) \Rightarrow (j \notin S), \quad \forall (i, j) \in E, \\ & D(S) = [\emptyset, V], 0 \leq |S| \leq |V|. \end{aligned}$$

Note that this model makes use of logical implications to ensure that no two adjacent vertices belong to the stable set.

### 22.2.2 Solving

The solution process of constraint programming interleaves *constraint propagation* and *search*. During the search process, all possible variable-value combinations are implicitly enumerated until we find a solution or prove that none exists. We say that this process constructs a *search tree*, in which the root node represents the original problem to be solved, and a branching decision partitions a node into at least two subproblems. To reduce the exponential number of combinations, *domain filtering* and *constraint propagation* is applied at each node of the search tree. A *domain filtering algorithm* operates on an individual constraint. Given a constraint, and the current domains of the variables in its scope, a domain filtering algorithm removes domain values that do not belong to a solution to the constraint. Such domain values are called *inconsistent* with respect to the constraint. Likewise, domain values that do belong to a solution are called *consistent* with respect to the constraint. Since variables usually participate in more than one constraint, the updated domains are propagated to the other constraints, whose domain filtering algorithms in effect become active. This process of constraint propagation is repeated for all constraints until no more domain values can be removed, or a domain becomes empty. Since the variable domains are finite, this process always terminates. Furthermore, it can be shown that under certain conditions on the domain filtering algorithms, the constraint propagation process always reaches the same fixed point, irrespective of the order in which the constraints are processed [3].

For example, consider the following constraint programming model (corresponding to a feasibility problem):

$$x_1 > x_2, \tag{22.1}$$

$$x_1 + x_2 = x_3, \tag{22.2}$$

$$\text{alldifferent}(x_1, x_2, x_3, x_4), \tag{22.3}$$

$$x_1 \in \{1, 2\}, x_2 \in \{0, 1, 2, 3\}, x_3 \in \{2, 3\}, x_4 \in \{0, 1\}. \tag{22.4}$$

If we process the constraints in the top-down order of the model, the constraint propagation first considers constraint (22.1), whose domain filtering algorithm removes the inconsistent values 2 and 3 from  $D(x_2)$ , i.e.,  $D(x_2) = \{0, 1\}$  at this point. We then continue with constraint (22.2), for which all values are consistent. For constraint (22.3), observe that variables  $x_2$  and  $x_4$  now both have domain  $\{0, 1\}$ , and thus saturate these values. Therefore, we can remove values 0 and 1 from the domains of the other variables in its scope, i.e.,  $D(x_1) = \{2\}$ . As an immediate effect, the same algorithm will deduce that  $D(x_3) = \{3\}$ . At this point, we have processed all constraints, but note that since we have changed the domains of  $x_1$ ,  $x_2$ , and  $x_3$ , we need to repeat the domain filtering for the constraints in which they participate. Considering again constraint (22.1), we find that all domain values are consistent. From constraint (22.2), however, we deduce that  $D(x_2) = \{1\}$ . Finally, reconsidering constraint (22.3) leads to  $D(x_4) = \{0\}$ . In summary, in this case the constraint propagation alone is able to find a solution to the problem, i.e.,  $x_1 = 2, x_2 = 1, x_3 = 3, x_4 = 0$ .

In order to be effective, domain filtering algorithms should be computationally efficient, because they are applied many times during the solution process. Further, they should remove as many inconsistent values as possible. If a domain filtering algorithm for a constraint  $C$  removes *all* inconsistent values from the domains with respect to  $C$ , we say that it makes  $C$  *domain consistent*.<sup>1</sup> In other words, all remaining domain values participate in at least one solution to  $C$ . More formally:

**Definition 22.1 (Domain consistency).** A constraint  $C$  on the variables  $x_1, \dots, x_k$  is called *domain consistent* if for each variable  $x_i$  and each value  $d_i \in D(x_i)$  ( $i = 1, \dots, k$ ), there exist a value  $d_j \in D(x_j)$  for all  $j \neq i$  such that  $(d_1, \dots, d_k) \in C$ .

In practice, one usually tries to develop filtering algorithms that separate the check for consistency and the actual domain filtering, especially when processing global constraints. That is, we would like to avoid applying the algorithm that performs the consistency check to each individual variable-value pair, since that strategy can often be improved. Moreover, one typically tries to design incremental algorithms that re-use data structures and partial solutions from one filtering event to the next, instead of applying the filtering algorithm from scratch each time it is invoked.

Without effective domain filtering algorithms to reduce the search space, constraint programming would not be applicable in practice. For many practical problems, global constraints are the most effective, since they exploit the underlying combinatorial structure to identify more inconsistent values than a corresponding decomposition in ‘elementary’ constraints would identify. For example, consider a set of variables  $X$  that must take pairwise different values, while the domain of each variable  $x \in X$  is  $D(x) = \{1, \dots, |X| - 1\}$ . Clearly, as the number of variables exceeds the number of possible values that can be jointly assigned to them, the problem has no solution. However, if we model the problem using not-equal constraints  $x_i \neq x_j$

---

<sup>1</sup>In the literature, domain consistency is also referred to as *hyper-arc consistency* or *generalized arc consistency* for historic reasons.

for all pairs  $x_i, x_j \in X$  with  $i \neq j$ , constraint propagation alone cannot deduce this fact. Namely, for each individual not-equal constraint  $x_i \neq x_j$ , the associated domain filtering algorithm will consider only the domains of  $x_i$  and  $x_j$ , and each value within these domains participates in a solution to that constraint. On the other hand, if we establish domain consistency for the global constraint `alldifferent`( $X$ ), we would be able to immediately deduce that the constraint cannot be satisfied. In fact, [56] showed that domain consistency for `alldifferent` can be established in polynomial time by representing the constraint as a bipartite matching problem.

As a final remark on domain filtering, consider the constraint  $(i \in S) \Rightarrow (j \notin S)$  as applied in the stable set model above. Such constraints, that involve the truth value of subexpressions, will internally be represented using Boolean variables by most CP systems. That is, one Boolean variable  $x$  represents whether the expression  $(i \in S)$  is true or not, and another Boolean variable  $y$  represents the expression  $(j \notin S)$ . The associated domain filtering will then be performed on the relation  $x \Rightarrow y$  and the relations  $x = (i \in S)$  and  $y = (j \notin S)$ . However, such decomposition into a Boolean representation is not always necessary for set variables, as more efficient algorithms may be implemented directly, especially for global constraints on set variables. For example, the global constraint `partition`( $S_1, S_2, \dots, S_n, V$ ) requires that the set variables  $S_1, S_2, \dots, S_n$  form a partition of the set  $V$ . For this constraint, an efficient filtering algorithm can be designed based on a bipartite network flow representation [59, 61].

When a constraint programming model involves an objective function to be optimized, the default solving process applies the following straightforward bounding procedure. Consider an objective  $\min f(X)$  where the objective function  $f$  can be an arbitrary expression mapped to a totally ordered set. Assuming that we have a known upper bound  $U$  on the objective, it will be treated as a constraint  $f(X) \leq U$ , and the associated domain filtering algorithms for this constraint will be applied. As soon as a solution satisfying all the constraints has been found, and that has objective value  $U'$ , the right hand side of the ‘objective constraint’ will be updated as  $f(X) \leq U' - 1$ , after which the search continues. Note that if the objective value of a partial solution is higher than the current upper bound, the associated domain filtering algorithm will detect an inconsistency and the solver will backtrack to a previous search state.

### 22.2.3 Optimization Constraints

As explained in the previous section, the CP solver evaluates the objective function as an individual constraint, where the potential solution set is formed by the Cartesian product of the domains of the variables in its scope. That is, by default it is not aware of any underlying combinatorial structure that may improve the bound of the objective, unless we add this structure explicitly to the constraint that represents the objective. Many works have studied how to improve the representation of the objective in constraint programming, and the resulting constraints are

now often referred to as ‘optimization constraints’, or ‘optimization oriented global constraints’ [14].

Even though a formal definition of optimization constraints is not easy to provide, they can be thought of as weighted versions of global constraints. That is, for a global constraint  $C(X)$  on a set of variables  $X$ , we can define a corresponding optimization constraint with respect to a ‘weight’ function  $f(X)$  as

$$C(X) \wedge (f(X) \leq z),$$

where  $z$  is a variable representing the upper bound on the weight function. The optimization constraint thus ensures that we only allow solutions that respect constraint  $C$  and have a corresponding weight not more than  $z$ . The domain filtering associated with such reasoning is referred to as ‘cost-based’ domain filtering. Cost-based domain filtering algorithms usually work in two directions: One direction is to update the domain of  $z$ , i.e., improve the bound, based on the current domains of the variables  $X$ , while the second direction is to update the domains of the variables  $X$  based on the current bound of  $z$ . The latter direction is also referred to as ‘back-propagation’ in the literature.

While many optimization constraints utilize specialized combinatorial algorithms to perform cost-based filtering (see, e.g., [38, 57, 62, 68]), a generic methodology to designing cost-based filtering for optimization constraints was introduced in a sequence of papers by Focacci et al. [17] and Focacci et al. [13, 15, 16]. In these works, the domain filtering is based on a linear relaxation of the associated global constraint together with a ‘gradient function’  $\text{grad}(x, v)$  that returns, for each possible variable-value pair  $(x, v)$  with  $v \in D(x)$ , an optimistic evaluation of the cost to be paid if  $v$  is assigned to  $x$ . For example, consider the constraint `alldifferent( $x_1, \dots, x_n$ )`, together with ‘weights’  $w_{ij}$  for every pair  $(x_i, j)$  such that  $j \in D(x_i)$  ( $i \in \{1, \dots, n\}$ ). We introduce binary variables  $y_{ij}$  for  $i \in \{1, \dots, n\}$  and  $j \in D(x_i)$  such that

$$\begin{aligned} y_{ij} = 1 &\Leftrightarrow (x_i = j), \text{ and} \\ y_{ij} = 0 &\Leftrightarrow (x_i \neq j). \end{aligned}$$

The weighted `alldifferent` constraint corresponds to the following Assignment Problem formulation:

$$\begin{aligned} z &= \sum_{i=1}^n \sum_{j \in D(x_i)} w_{ij} y_{ij} \\ \sum_{j \in D(x_i)} y_{ij} &= 1, \quad \forall i \in \{1, \dots, n\}, \\ \sum_{i=1}^n y_{ij} &\leq 1, \quad \forall j \in \cup_{i=1}^n D(x_i), \\ y_{ij} &\in \{0, 1\}, \quad \forall i \in \{1, \dots, n\}, \forall j \in D(x_i), \end{aligned}$$

where  $z$  represents the value of the objective function. It is well-known that the linear relaxation of the Assignment Problem yields an integer solution. Focacci et al. [13] propose to use the reduced costs obtained by solving this relaxation as the gradient function. That is, let  $\bar{c}_{ij}$  represent the reduced cost of variable  $y_{ij}$  in a solution to the Assignment Problem relaxation with lower bound  $z^*$ , and let  $z_{\min}$  be the current best solution value. Whenever  $z^* + \bar{c}_{ij} > z_{\min}$ , we can remove value  $j$  from  $D(x_i)$ . Note that in the Operations Research literature this technique is known under the name ‘variable fixing’ [50]. This approach has been generalized to Lagrangian relaxations, where the gradient function is formed by the Lagrange multipliers, see for example [34].

## 22.2.4 Search Strategies

Even though the search process of constraint programming has similarities with integer programming, there are several differences. The goal of the search process is to implicitly enumerate all possible variable-value combinations until a (provably optimal) solution has been found, or until it is proved that no solution exists. As stated before, this process is said to create a search tree, which is a particular rooted tree. The root node represents the original problem to be solved. At each node of the tree constraint propagation is applied, typically until a fixed point is reached. If this process does not prove infeasibility, and does not finish with a solution, the node is split into two or more subproblems. We must ensure that the union of the solution set of the subproblems is equal to the solution set of the problem represented by the node itself.

The splitting, or branching, procedure can take various forms, and an important difference with integer programming systems is that a user can define the branching decisions to be made. In many cases this can even be done at the modeling level, see, e.g., Van Hentenryck et al. [64]. A branching decision is stated as a constraint, which will be added to the subproblem (its negation will be added to the other subproblem in case of a binary branching decision). For example, for a variable  $x$  with  $v \in D(x)$ , we can branch on the constraint  $(x = v)$  versus  $(x \neq v)$ . This corresponds to the traditional enumeration of variable-value combinations.

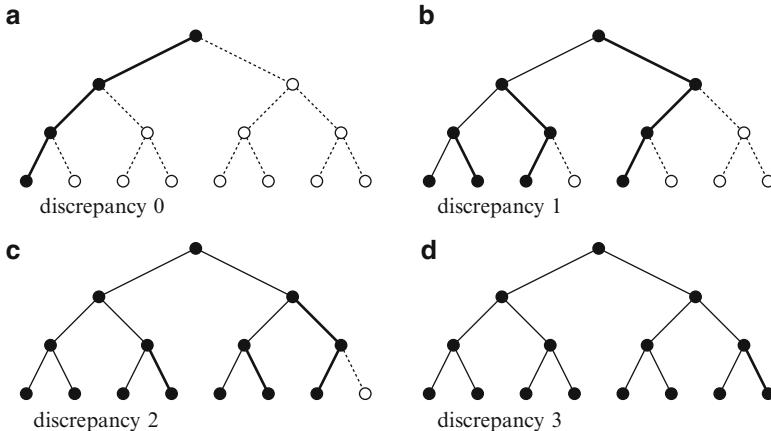
In order to apply a particular branching constraint, we first need to identify one or more variables on which to define the constraint. When the branching decisions correspond to variable assignments  $(x = v)$  versus  $(x \neq v)$ , the order in which variables, and corresponding values, are considered follows from so-called *variable selection* heuristics and *value selection* heuristics. That is, when the problem is not infeasible and one or more variables are still ‘free’ (i.e., they have a domain with size larger than one), the variable selection heuristic will identify one variable  $x$  from the set of free variables. Similarly, the value selection heuristic will identify one value  $v$  from its domain  $D(x)$ . A standard heuristic in constraint programming

is to choose first the variable with the smallest domain size (this is called fail-first), and assign it the smallest value in its domain. For more involved heuristics, we refer to [63].

In addition to the branching decisions that define the shape of the search tree, the search process also depends on the order in which the different search tree nodes are visited. In integer programming, the order is almost always defined by some variant of a ‘best-bound-first’ strategy, in which the next node to explore is that with the most promising bound following from the linear relaxation. Even though this strategy is often superior in practice for integer programming problems, it has the potential drawback of creating an exponential number of active search nodes. Here, a node is considered active when it must be split further, i.e., it is not proved to be infeasible or suboptimal and it does not correspond to a solution. Best-bound-first and similar strategies can also be applied to constraint programming search trees. However, it is much more common to apply a depth-first search strategy instead. An important reason for this may be the fact that constraint programming does not contain ‘primal heuristics’ that can find feasible solutions quickly for a given search node. In fact, one may view the constraint programming search process itself as a primal heuristic. Furthermore, the memory consumption of depth-first is linear in the depth of the search tree, while this can be exponential for breadth-first search strategies (including best-bound-first).

Search strategies have been studied extensively in the context of constraint programming, and more generally in the context of artificial intelligence; see van Beek [63] for an overview. One particular search strategy that will also be applied in this chapter is *limited discrepancy search*, introduced by Harvey and Ginsberg [31]. It is based on the assumption that we have a good, but not perfect, branching heuristic available (e.g., corresponding to a variable and value selection heuristic). If the branching heuristic were perfect, the first visited leaf in the search tree (when applying depth-first search) would correspond to a solution to the problem (assuming it exists). Harvey and Ginsberg [31] argue that when the heuristic is equally likely to make a mistake at each search node, we should visit the leafs by increasing value of total discrepancy from the heuristic choice required to reach that leaf. An illustration is given in Fig. 22.1. In the binary search tree of this figure, it is assumed that the left branch corresponds to the heuristic choice. Discrepancy 0 (in Fig. 22.1a thus corresponds to the first leaf found by depth-first search. The leafs visited for discrepancy 1 (in Fig. 22.1b correspond to all leafs that are reached when exactly one right branch is chosen, and so on.

Note that the ‘Local Branching’ search strategy in the context of integer programming resembles limited discrepancy search [11]. Namely, given a current solution  $x^*$  and an integer  $k$ , Local Branching creates a subproblem in which  $k$  variables are allowed to deviate from the value taken in  $x^*$ . The solutions considered in that subproblem correspond to those having discrepancy 0 up to  $k$ , where  $x^*$  serves as the heuristic to be followed.



**Fig. 22.1** Limited discrepancy search. For each discrepancy 0, 1, 2, 3, the top node of the tree indicates the root, visited nodes are filled, and bold arcs indicate the parts of the tree that are traversed newly. (a) discrepancy 0. (b) discrepancy 1. (c) discrepancy 2. (d) discrepancy 3

## 22.3 Semidefinite Relaxations in Constraint Programming

There exist two streams of research that combine semidefinite relaxations and constraint programming. The first stream considers constraint programming models of an arbitrary form, and assumes that any suitable semidefinite relaxation may be applied, depending on the problem at hand. It is in the same spirit as the ‘optimization constraints’ mentioned before, and this stream will be the main focus of the remainder of this chapter. The second stream considers constraint programming models of a specific form, and applies a dedicated semidefinite relaxation that maps exactly onto the format of the constraint programming model. Examples are given at the end of this section.

In this section, we will first discuss how suitable semidefinite relaxations may be found for arbitrary constraint programming models. Unfortunately, it is not straightforward to obtain a computationally efficient semidefinite program that provides a tight solution for any given constraint programming model. However, for a number of combinatorial optimization problems such semidefinite relaxations do exist, see Laurent and Rendl [44] and Chap. 27 of this Handbook for an overview. If such semidefinite relaxations are not available, we need to build up a relaxation from scratch. One possible way is to apply the generic framework by Laurent et al. [43] that will be described in Sect. 22.3.1.

After the discussion of building semidefinite relaxations, we consider the application of such relaxations inside optimization constraints in Sect. 22.3.2. Next, we discuss how the solution to the semidefinite relaxation can be applied to guide the search process in Sect. 22.3.3. Finally, in Sect. 22.3.4 we discuss other applications to combining semidefinite relaxations and constraint programming, that are in the second stream mentioned above.

### 22.3.1 Building a Semidefinite Relaxation

Consider a constraint programming model consisting of a set of variables  $x_1, \dots, x_n$ , a set of constraints and an objective function. If the domains  $D(x_1), \dots, D(x_n)$  are not binary, we first transform the variables and their domains into corresponding binary variables  $y_{ij}$  for  $i = 1, \dots, n$  and  $j \in D(x_i)$ :

$$\begin{aligned} x_i = j &\Leftrightarrow y_{ij} = 1, \\ x_i \neq j &\Leftrightarrow y_{ij} = 0. \end{aligned} \tag{22.5}$$

We will use the binary variables  $y_{ij}$  to construct a semidefinite relaxation, following Laurent et al. [43]. Of course, the transformation has consequences for the constraints also, which will be discussed below. The method of Laurent et al. [43] to transform a model with binary variables into a semidefinite relaxation is the following. For a positive integer  $N$ , let  $\mathbf{d} \in \{0, 1\}^N$  be a vector of binary variables representing all variables  $y_{ij}$  for  $i = 1, \dots, n$  and  $j \in D(x_i)$ . Construct the  $(N+1) \times (N+1)$  variable matrix  $X$  as

$$X = \begin{pmatrix} 1 \\ \mathbf{d} \end{pmatrix} \begin{pmatrix} 1 & \mathbf{d}^\top \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{d}^\top \\ \mathbf{d} & \mathbf{d}\mathbf{d}^\top \end{pmatrix}. \tag{22.6}$$

We then constrain  $X$  to satisfy

$$X \succeq 0 \tag{22.7}$$

$$X_{ii} = X_{0i} \quad \forall i \in \{1, \dots, N\} \tag{22.8}$$

where the rows and columns of  $X$  are indexed from 0 to  $N$ . Condition (22.8) expresses the fact that  $d_i^2 = d_i$ , which is equivalent to  $d_i \in \{0, 1\}$ . Note however that the latter constraint is relaxed by requiring  $X$  to be positive semidefinite.

The matrix  $X$  contains the variables to model our semidefinite relaxation. Obviously, the diagonal entries (as well as the first row and column) of this matrix represent the binary variables from which we started. Using these variables, we need to express (a part of) the original constraints as our semidefinite relaxation.

In case the binary variables are obtained from transformation (22.5), not all constraints may be trivially transformed accordingly. Especially because the original constraints may be of any form. The same holds for the objective function. On the other hand, as we are constructing a relaxation, we may choose among the set of constraints an appropriate subset to include in the relaxation. Moreover, the constraints itself are allowed to be relaxed.

In most practical cases, however, the suitability of applying an SDP relaxation heavily relies on the problem at hand. Instead of applying the above framework, a practitioner may be more interested in identifying a combinatorial substructure of the problem for which a known SDP relaxation exists and can be readily applied.

### 22.3.2 Semidefinite Relaxation as Optimization Constraint

Similar to linear programming relaxations, semidefinite programming relaxations can be embedded inside optimization constraints. As discussed before, the role of such a constraint is twofold. First, the relaxation can be used to improve the bound on the variable representing the corresponding objective. Second, the relaxation can be applied to perform cost-based domain filtering.

So far, semidefinite relaxations have only been applied in a constraint programming context to improve the bound on the objective. Cost-based domain filtering utilizing the semidefinite relaxation has not yet been pursued. Nevertheless, Helmberg [32] has introduced variable fixing procedures for semidefinite programming, while Fleischman and Poggi de Aragão [12] have presented similar variable fixing procedures for unconstrained quadratic programs. In principle, these methods could be readily applied as filtering algorithms inside an optimization constraint. However, there are some practical hurdles that may prevent such filtering to be worthwhile. Perhaps the most important one is the issue of incrementality. Recall that domain filtering algorithms are typically invoked many times during the solving process. Since it is not straightforward to efficiently re-use data from one propagation event to the next when using semidefinite programming, the relaxations typically have to be recomputed from scratch at each event. This can be very time-consuming, and may not offset the extra amount of search space reduction that can be gained from it.

### 22.3.3 Semidefinite Relaxation as Search Heuristic

Several works on optimization constraints have applied the embedded linear relaxation not only for cost-based filtering, but also to define a search heuristic. For example, Focacci et al. [13] use the solution to the linear relaxation as a branching heuristic for Traveling Salesman Problems. As another example, Milano and Van Hoeve [49] apply reduced costs following from a linear relaxation as a search heuristic. Leahu and Gomes [45] investigate the quality of linear relaxations as a search heuristic in detail.

Van Hoeve [65, 66] proposes to use the solution to an SDP relaxation as a heuristic to guide the CP search. In general, the solution to a semidefinite relaxation yields fractional values for its variable matrix. These fractional values can serve as an indication for the values that the original constraint programming variables take in a solution. Consider for example the matrix  $X$  of equation (22.6) above, and suppose it is obtained from non-binary original variables by transformation (22.5). Assume that variable  $X_{ii}$  corresponds to the binary variable  $y_{jk}$  (for some integer  $j$  and  $k$ ), which corresponds to  $x_j = k$ , where  $x_j$  is a constraint programming variable and  $k \in D(x_j)$ . If variable  $X_{ii}$  is close to 1, then also  $y_{jk}$  is supposed to be close to 1, which corresponds to assigning  $x_j = k$ .

Hence, the variable and value selection heuristics for the constraint programming variables are based upon the fractional solution values of the corresponding variables in the semidefinite relaxation. A natural variable selection heuristic is to select first the constraint programming variable for which the corresponding variable in the semidefinite relaxations has a fractional solution that is closest to the corresponding integer solution. As value selection heuristic, we then select first the corresponding suggested value. As an alternative, we can design a randomized branching heuristic, in which a selected variable (or value) is accepted with probability proportional to the corresponding fractional value in the semidefinite relaxation.

Note that not all semidefinite relaxations may offer a precise mapping between a variable-value pair in the constraint programming model and a variable in the semidefinite relaxation. We will see an example of this in Sect. 22.6, where the constraint programming variables  $x_1, \dots, x_n$  all have domains  $\{1, \dots, k\}$  for some integer  $k \geq 1$ . The semidefinite relaxation of that application uses variables  $Z_{ij}$  that represent whether  $x_i$  and  $x_j$  are assigned the same value, irrespective of the eventual value. In such situations, we can still apply the solution to the semidefinite relaxation as a branching heuristic, for example by branching on  $(x_i = x_j)$  versus  $(x_i \neq x_j)$ . However, additional branching decisions may be necessary to eventually assign a value to each variable.

### 22.3.4 Semidefinite Relaxations for Restricted Constraint Programming Problems

Several works have applied semidefinite relaxations to constraint programming problems of a specific form, including Boolean satisfiability and weighted constraint satisfaction problems (CSPs). The *Boolean satisfiability problem*, or SAT problem, consists of a set of Boolean variables and a conjunction of clauses, each of which is a disjunction of literals. The conjunction of clauses is called a formula. Each literal is a logical variable ( $x$ ) or its negation ( $\bar{x}$ ). The SAT problem is to determine whether there exists a variable assignment that makes the formula true (i.e., each clause is true). The  $k$ -SAT problem represents the SAT problem where the clauses are constrained to have length equal to  $k$ . The  $(2+p)$ -SAT problem is a SAT problem in which a fraction  $p$  of the clauses is defined on three literals, while a fraction  $(1-p)$  is defined on two literals. Clearly, the SAT problem is a constraint programming problem of a restricted form, i.e., the variables take Boolean domains and constraints are restricted to take the form of clauses.

MAX-SAT is the optimization version of SAT. Given a formula, we want to maximize the number of simultaneously satisfied clauses. Given an algorithm for MAX-SAT we can solve SAT, but not vice-versa, therefore MAX-SAT is more complex than SAT. The distinction becomes obvious when considering the case when the clauses are restricted to two literals per clause (2-SAT): 2-SAT is solvable in linear time, while MAX-2-SAT is NP-hard [19].

Goemans and Williamson [23] were the first to apply semidefinite relaxations to the MAX-2-SAT and MAX-SAT problems for obtaining approximation algorithms with provable performance ratios. Other works following this approach include [37] for MAX-3-SAT and [30] for MAX-4-SAT. Warners [69] applies semidefinite relaxations for MAX-2-SAT problems within a DPLL-based SAT solver (see also [40]). The experimental results show that the semidefinite relaxations provide very tight objective values, and are applicable in practice in terms of computational efficiency. De Klerk et al. [41] study semidefinite relaxations for general SAT problems, while De Klerk and Van Maaren [39] consider semidefinite relaxations for  $(2+p)$ -SAT problems. Anjos [1] introduces improved semidefinite relaxations to SAT problems.

Lau [42] considers *weighted CSPs*, that generalize the MAX-2-SAT problem above in two ways. First, variables can have arbitrary finite domains. Second, even though the constraints are still defined on two variables only, now a weight is associated to each constraint. That is, a constraint  $C$  on two variables  $x$  and  $y$  is defined as  $C \subseteq D(x) \times D(y)$ , and has an associated weight (a natural number). The goal is to maximize the sum of the weights of the satisfied constraints. Lau [42] represents such weighted CSPs as a quadratic integer program, and formulates a corresponding semidefinite relaxation. He applies a randomized rounding procedure to prove worst-case bounds for fixed variable domain sizes 2 and 3. He also provides computational results, comparing his approach to greedy and randomized local search procedures.

A recent line of research studies problems of the form MAX  $k$ -CSP, where variables can take an arbitrary finite domain, each constraint is defined on  $k$  variables, and the goal is to maximize the number of satisfied constraints. Also Zwick [71] applies semidefinite relaxations to a similar problem referred to as MAX 3-CSP, but in that case the domains are Boolean, constraints take the form of a Boolean function on at most three variables, and each constraint has an associated weight. Therefore, these problems are more similar to weighted CSPs than to the recent interpretation of MAX  $k$ -CSP. Most of the recent works for MAX  $k$ -CSP consider the approximability of such problems, and apply semidefinite relaxations to obtain certain approximation ratios, e.g., [6, 29] and [52]. Other works study the integrality gap following from the semidefinite relaxation, for similar constraint satisfaction problems, e.g., [53, 54].

## 22.4 Application to the Maximum Weighted Stable Set Problem

As a first application of integrating constraint programming and semidefinite programming we consider the hybrid approach of Van Hoeve [65, 66] for the stable set problem. The motivation for this work is based on two complementary observations: (a) a standard CP approach can have great difficulty finding a good solution, let alone proving optimality, and (b) SDP relaxations may provide a good starting solution, but the embedding of SDP inside a branch-and-bound framework to prove optimality can be computationally too expensive.

### 22.4.1 Problem Description and Model Formulations

Recall from Sect. 22.2 that the weighted stable set problem is defined on an undirected weighted graph  $G = (V, E)$ , with ‘weight’ function  $w : E \rightarrow \mathbb{R}$ . Without loss of generality, we assume all weights to be non-negative. The problem is to find a subset of vertices  $S \subseteq V$  of maximum total weight, such that no two vertices in  $S$  are joined by an edge in  $E$ .

The constraint programming model applied in [66] uses binary variables  $x_i$  representing whether vertex  $i$  is in  $S$  ( $x_i = 1$ ) or not ( $x_i = 0$ ), for all  $i \in V$ . The CP model is then formulated as an integer linear programming model:

$$\begin{aligned} \max \quad & \sum_{i=1}^n w_i x_i \\ \text{s.t. } & x_i + x_j \leq 1, \forall (i, j) \in E, \\ & x_i \in \{0, 1\}, \forall i \in V. \end{aligned} \tag{22.9}$$

A, by now classical, semidefinite relaxation for the maximum-weight stable set problem was introduced by Lovász [46]. The value of that relaxation is called the theta number, and is denoted by  $\vartheta(G)$  for a graph  $G$ . The theta number arises from several different formulations, see Grötschel et al. [25]. Van Hoeve [66] uses the formulation that has been shown to be computationally most efficient among those alternatives [26]. Let us introduce that particular formulation (called  $\vartheta_3$  by Grötschel et al. [25]). Let  $\mathbf{x} \in \{0, 1\}^n$  be the vector of binary variables representing a stable set, where  $n = |V|$ . Define the  $n \times n$  matrix  $X = \xi \xi^\top$  where the vector  $\xi$  is given by

$$\xi_i = \frac{\sqrt{w_i}}{\sqrt{\sum_{j=1}^n w_j}} x_j$$

for all  $i \in V$ . Furthermore, let the  $n \times n$  cost matrix  $U$  be defined as  $U_{ij} = \sqrt{w_i w_j}$  for  $i, j \in V$ . Observe that in these definitions we exploit the fact that  $w_i \geq 0$  for all  $i \in V$ . The following semidefinite program

$$\begin{aligned} \max \quad & \text{tr}(UX) \\ \text{s.t. } & \text{tr}(X) = 1 \\ & X_{ij} = 0, \forall (i, j) \in E \\ & X \succeq 0 \end{aligned} \tag{22.10}$$

provides the theta number of  $G$ , see [25]. Here  $\text{tr}(X)$  represent the trace of matrix  $X$ , i.e., the sum of its main diagonal elements. When (22.10) is solved to optimality, the diagonal element  $X_{ii}$  can be interpreted as an indication for the value that  $x_i$  ( $i \in V$ ) takes in an optimal solution to the problem.

### 22.4.2 Evaluation of the Hybrid Approach

In the hybrid approach of Van Hoeve [66], the semidefinite relaxation is solved once, at the root node of the search tree. The associated objective value is applied to bound the initial domain of the variable representing the objective in the CP model. Van Hoeve [66] does not apply any additional cost-based domain filtering. Instead, the fractional solution to the semidefinite relaxation is applied as a variable and value selection heuristic. The variable  $x_i$  with the highest corresponding solution for  $X_{ii}$  will be selected first, and value 1 will be assigned to it first. Since the semidefinite relaxation often provides a very accurate variable-value selection heuristic, in [66] limited discrepancy search is applied to traverse the resulting search tree.

This approach is applied to solve random problem instances as well as structured instances arising from coding theory, and maximum clique problems. The hybrid approach is compared to a pure CP solver with a lexicographic variable selection strategy, choosing value 1 first. In almost all cases, the SDP solution provides a very accurate branching strategy, and the best solution is found at a very low discrepancy (recall that limited discrepancy search is applied). In fact, in many cases the tight bound obtained by the SDP relaxation suffices to prove optimality of the solution found with the SDP-based heuristic.

## 22.5 Accuracy of Semidefinite Relaxations as Search Heuristic

In this section, the accuracy of the semidefinite relaxation as a search heuristic is investigated in more detail. A similar investigation has been performed for linear programming relaxations by Leahu and Gomes [45]. They identify that the heuristic quality of the LP solution is dependent on structural combinatorial properties of the problem at hand, which in their experiments is measured by the ‘constrainedness’ of the problem. More specifically, the problems that they consider, i.e., Latin Square completion problems, exhibit an easy-hard-easy phase transition when the problem becomes more constrained. The ‘typically hardest’ problem instances are those that originate from the critically constrained region corresponding to the easy-hard-easy phase transition. For instances that are outside this region, and that are typically less hard to solve, the linear programming relaxation provides quite accurate values. However, for problem instances from within the phase transition region, the information quality as search heuristic shows a sharp decrease. In other words, the quality of the relaxation degrades exactly for those instances it is most needed.

In light of these results, Gomes et al. [24] study the accuracy of semidefinite relaxations as search heuristic. One of the main motivations for this was to investigate whether semidefinite relaxations provide more robust search heuristics, and of higher quality, than linear programming relaxations. The particular problem

of study in [24] is MAX-2-SAT, and SDP relaxations are contrasted with LP relaxations and complete (exact) solution methods for this problem.

A related work is that of Warners [69] and De Klerk and Warners [40], who propose and analyze a MAX-2-SAT solver that employs a semidefinite relaxation as well. However, they do not apply the SDP solution as a search heuristic. Instead, the branching rule is to choose first the variable with the maximal occurrence in the longest clauses. De Klerk and Van Maaren [39] present another related work, in which the accuracy of the semidefinite relaxation to detect unsatisfiability for the  $(2+p)$ -SAT problem is experimentally evaluated. Finally, we note that Cherian et al. [7] also apply linear programming and rounding techniques to solve MAX-2-SAT problems.

### 22.5.1 Problem Description and Model Formulations

Let the MAX-2-SAT problem consist of Boolean variables  $x_1, x_2, \dots, x_n$  and clauses  $C_1, C_2, \dots, C_m$  on these variables. We consider the following ILP formulation for the MAX-SAT problem from Goemans and Williamson [22]. With each clause  $C_j$  we associate a variable  $z_j \in \{0, 1\}$ , for  $j = 1, \dots, m$ . Value 1 corresponds to the clause being satisfied and 0 to the clause not being satisfied. For each Boolean variable  $x_i$  we associate a corresponding variable  $y_i$  in the ILP, for  $i = 1, \dots, n$ . Variable  $y_i$  can take the values 0 and 1, corresponding to  $x_i$  being false or true, respectively. Let  $C_j^+$  be the set of indices of positive literals that appear in clause  $C_j$ , and  $C_j^-$  be the set of indices of negative literals (i.e., complemented variables) that appear in clause  $C_j$ . The problem can then be stated as follows:

$$\begin{aligned} & \max \sum_{j=1}^m z_j \\ \text{subject to } & \sum_{i \in C_j^+} y_i + \sum_{i \in C_j^-} (1 - y_i) \geq z_j, \quad \forall j \in \{1, \dots, m\} \\ & y_i, z_j \in \{0, 1\}, \quad \forall i \in \{1, \dots, n\}, j \in \{1, \dots, m\}. \end{aligned}$$

This model ensures that a clause is true only if at least one of the variables that appear in the clause has the value 1. Since we maximize  $\sum_{j=1}^m z_j$  and  $z_j$  can be set to 1 only when clause  $C_j$  is satisfied, it follows that the objective function counts the number of satisfied clauses. By relaxing the integrality constraint, we obtain an LP relaxation for the MAX-SAT problem. This ILP formulation is equivalent to the ILP used in [70] to compute the lower bound and to the ILP solved at each node by the MAX-SAT branch-and-cut algorithm in [36].

Observe that there exists a trivial way to satisfy all the clauses by setting each variable  $y_i$  to 0.5. Using this assignment, the sum of literals for each clause is exactly 1, hence the clause can be satisfied and the objective function is equal to the number of clauses. The value 0.5 is not at all informative, lying half way between 0 and 1,

it gives no information whether the corresponding Boolean variable should be set to true or false. As the problem becomes more constrained (i.e., the number of clauses increases) the corresponding 2-SAT problem is very likely to be unsatisfiable, hence any variable assignment different than 0.5 would lead to a less than optimal objective value. Naturally, the LP solver finds the highest possible objective value (i.e., the number of clauses) when setting all variables to 0.5.

Gomes et al. [24] apply the following semidefinite relaxation of MAX-2-SAT that was introduced by Goemans and Williamson [23]. To each Boolean variable  $x_i$  ( $i = 1, \dots, n$ ), we associate a variable  $y_i \in \{-1, 1\}$ . Moreover, we introduce a variable  $y_0 \in \{-1, 1\}$ . We define  $x_i$  to be true if and only if  $y_i = y_0$ , and false otherwise.

Next, we express the truth value of a Boolean formula in terms of its variables. Given a formula  $c$ , we define its *value*, denoted by  $v(c)$ , to be 1 if the formula is true, and 0 otherwise. Hence,

$$v(x_i) = \frac{1 + y_0 y_i}{2}$$

gives the value of a Boolean variable  $x_i$  as defined above. Similarly,

$$v(\bar{x}_i) = 1 - v(x_i) = \frac{1 - y_0 y_i}{2}.$$

The value of the formula  $x_i \vee x_j$  can be expressed as

$$\begin{aligned} v(x_i \vee x_j) &= 1 - v(\bar{x}_i \wedge \bar{x}_j) = 1 - v(\bar{x}_i)v(\bar{x}_j) = 1 - \frac{1 - y_0 y_i}{2} \frac{1 - y_0 y_j}{2} \\ &= \frac{1 + y_0 y_i}{4} + \frac{1 + y_0 y_j}{4} + \frac{1 - y_i y_j}{4}. \end{aligned}$$

The value of other clauses can be expressed similarly. If a variable  $x_i$  is negated in a clause, then we replace  $y_i$  by  $-y_i$  in the above expression.

Now we are ready to state the integer quadratic program for MAX-2-SAT:

$$\begin{aligned} \max \quad & \sum_{c \in C} v(c) \\ \text{s.t. } & y_i \in \{-1, 1\} \quad \forall i \in \{0, 1, \dots, n\}. \end{aligned} \tag{22.11}$$

It is convenient to rewrite this program as follows. We introduce an  $(n+1) \times (n+1)$  matrix  $Y$ , such that entry  $Y_{ij}$  represents  $y_i y_j$  (we index the rows and columns of  $Y$  from 0 to  $n$ ). Then program (22.11) can be rewritten as

$$\begin{aligned} \max \quad & \text{tr}(WY) \\ \text{s.t. } & Y_{ij} \in \{-1, 1\} \quad \forall i, j \in \{0, 1, \dots, n\}, i \neq j, \end{aligned} \tag{22.12}$$

where  $W$  is an  $(n+1) \times (n+1)$  matrix representing the coefficients in the objective function of (22.11). For example, if the coefficient of  $y_i y_j$  is  $w_{ij}$ , then  $W_{ij} = W_{ji} = \frac{1}{2}w_{ij}$ .

The final step consists in relaxing the conditions  $Y_{ij} \in \{-1, 1\}$  by demanding that  $Y$  should be positive semidefinite and  $Y_{ii} = 1 \forall i \in \{0, 1, \dots, n\}$ . The semidefinite relaxation of MAX-2-SAT can then be formulated as

$$\begin{aligned} & \max \operatorname{tr}(WY) \\ \text{s.t. } & Y_{ii} = 1 \quad \forall i \in \{0, 1, \dots, n\}, \\ & Y \geq 0. \end{aligned} \tag{22.13}$$

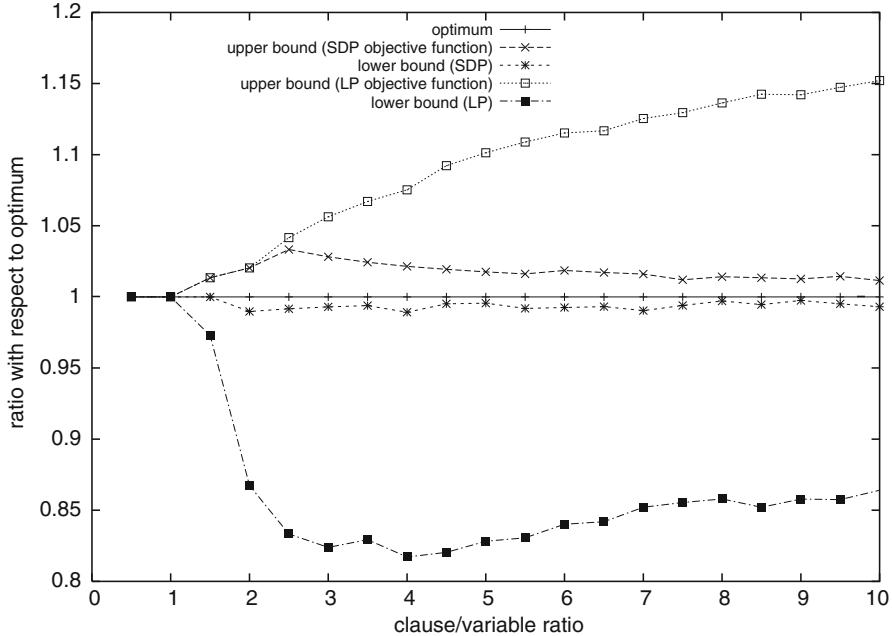
Program (22.13) provides an upper bound on the solution to MAX-2-SAT problems. Furthermore, the values  $Y_{0i}$ , representing  $y_0 y_i$ , correspond to the original Boolean variables  $x_i$  ( $i = 1, \dots, n$ ). Namely, if  $Y_{0i}$  is close to 1, variable  $x_i$  is ‘likely’ to be true. Similarly, if  $Y_{0i}$  is close to  $-1$ , variable  $x_i$  is ‘likely’ to be false.

### 22.5.2 Experimental Evaluation

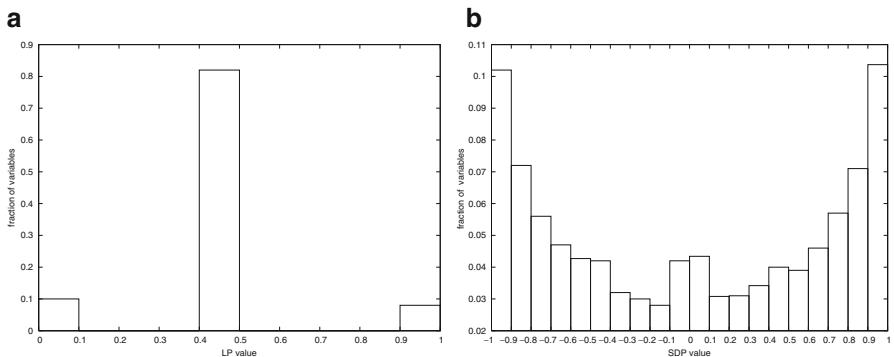
We next describe the experiments performed by Gomes et al. [24] on randomly generated MAX-2-SAT instances. First, we consider the objective value of the LP and SDP relaxations across different constrainedness regions of the problem. Figure 22.2 presents the relative lower and upper bounds obtained by LP and SDP, for MAX-2-SAT instances on 80 variables where the constrainedness ratio (number of clauses over the number of variables) ranges from 0.5 to 10. The figure compares the ratio of these bounds to the optimal solution value. The lower bounds for LP and SDP are obtained by rounding the suggested fractional solution to the closest integer. As is clear from this plot, the semidefinite relaxation provides far better bounds than the linear relaxation. This confirms the observations made by Warners [69] for MAX-2-SAT problems.

Next we consider the strengths of the relaxations in terms of heuristic guidance. This is done by first measuring the ‘fractionality’ of the solution to the relaxation. Namely, if a solution value of the relaxation is in the midpoint between the two integer endpoints (i.e., 0.5 for LP and 0.0 for SDP), we consider the suggested value uninformative. On the other hand, if the suggested value is close to an integer endpoint, we consider that value informative.

Figure 22.3 presents the distribution of the solution values returned by the LP and SDP relaxations, respectively. In Fig. 22.3a the distributions for the LP relaxation is given, clearly indicating that most values (more than 80%) are uninformative. In contrast, the distribution of the solution values for the SDP relaxation indicates that the majority of the suggested values is close to an integer value, and are much more informative. In this figure, the distribution averages the solutions over all instances where the ratio of the number of clauses over the number of variables ranges from 0.5 to 10.

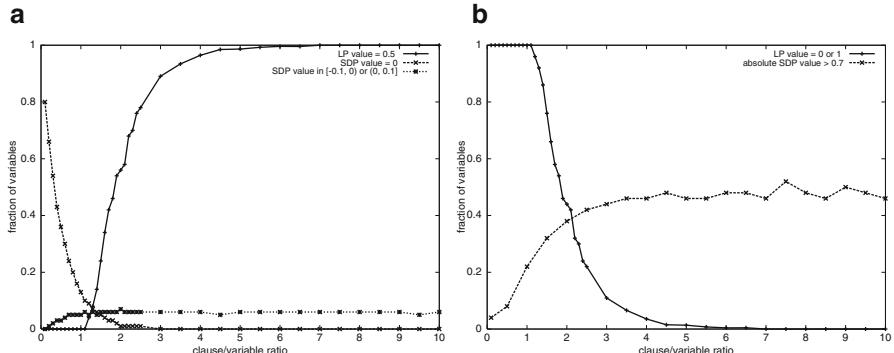


**Fig. 22.2** Lower and upper bounds based on LP and SDP relaxations for MAX-2-SAT instances of increasing constrainedness

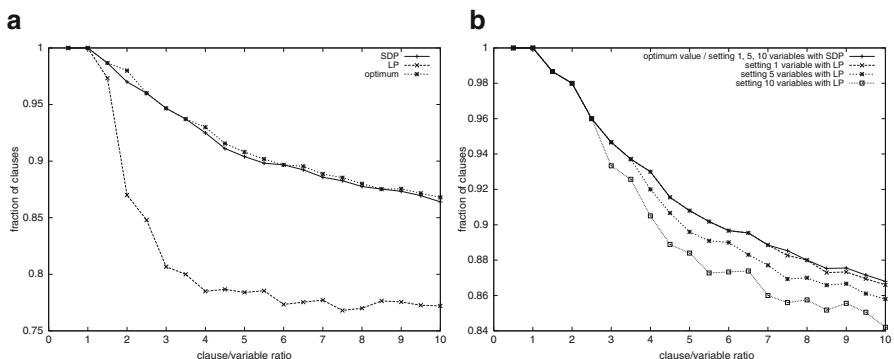


**Fig. 22.3** Distribution of the values returned by the LP relaxation (a) and the SDP relaxation, (b) averaged over instances with clause over variable ratio ranging from 0.5 to 10

Figure 22.4 depicts how the fractionality evolves with respect to the range of increasing constrainedness. In Fig. 22.4a the uninformative values are considered. It depicts the fraction of variables taking value 0.5 in the LP solution, and value 0, respectively in the intervals close to 0, i.e.,  $[-0.1, 0]$ ,  $(0, 0.1]$ , for the SDP solution. When the number of clauses increases (clause/variable ratio greater than 1), the LP solution values quickly become more uninformative. The SDP solution values of 0



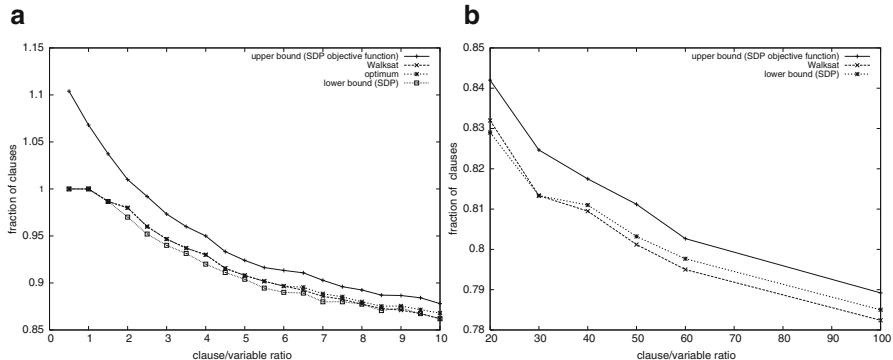
**Fig. 22.4** Fraction of variables having (a) uninformative value 0.5 computed by LP and 0 or smaller than 0.1 in absolute value computed by SDP, and (b) informative value 0 or 1 computed by the LP relaxation and above 0.7 in absolute value by the SDP relaxation



**Fig. 22.5** Change in the number of satisfied clauses as we set (a) 84% of the variables and (b) 1, 5 and 10 variables using the LP and SDP solutions

show exactly the opposite, while the SDP values close to 0 remain fairly constant. In Fig. 22.4b considers the informative values, i.e., LP solution values 0 or 1, and SDP values close to -1 or 1 (with absolute value more than 0.7 to be precise). Again, the LP solution values quickly loose their informative quality beyond clause/variable ratio 1. At the same time, the informative quality of the SDP solution values increases up to that point, and remains constant afterward.

The following set of experiments investigate the actual accuracy of the heuristics, by evaluating the optimal objective function value when a number of variables is fixed to the suggested heuristic value, where the variables closest to integrality are chosen to be fixed first. In Fig. 22.5a, 84% of the variables are fixed according to the heuristic, and the exact solver Maxsolver is used to compute an optimal solution, completing the partial assignment. Below 84%, the SDP heuristic choice always provided an optimal solution, while at 84%, the SDP heuristic



**Fig. 22.6** Comparing SDP and Walksat as a lower bound. **(a)** small clause/variable ratio. **(b)** large clause/variable ratio

deviates slightly from the optimal solution. The LP solution values perform much worse, as is clear from the figure. In Fig. 22.5b, this is shown in more detail, as it indicates the effect of setting 1, 5, and 10 variables to the suggested heuristic value. Even when assigning only one variable, the LP heuristic already deviates from the optimal solution for higher clause/variable ratios.

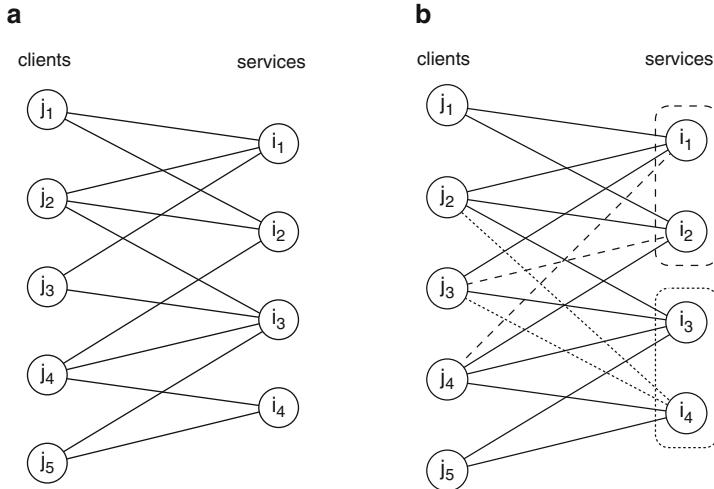
Finally, the solution obtained by the SDP heuristic is compared to the solution obtained by Walksat, a state-of-the-art satisfiability solver based on local search. The results are indicated in Fig. 22.6. It can be observed that Walksat and the SDP heuristic provide comparable solutions, and when the clause/variable ratio is relatively low (Fig. 22.6a), Walksat usually finds slightly better solutions than the heuristic guided by the SDP solution. For larger clause/variable ratio, however, the solutions obtained with the SDP heuristic outperform Walksat. This is an interesting result, because for these problems, the time to solve the SDP relaxation was considerably smaller than the time taken by Walksat, while at the same time the SDP relaxation also provides an upper bound.

In summary, these results further motivate the use of semidefinite relaxations as search heuristic for exact solving methods such as constraint programming.

## 22.6 Telecommunications Application: Biclique Completion

Gualandi [27] presents a hybrid semidefinite and constraint programming approach to a particular ‘biclique completion problem’, arising in the context of telecommunications. Let us first describe the application, which was first introduced by Faure et al. [9].

Traditional communication services are mostly ‘unicast’, in which two nodes of a network are communicating between each other (for example a telephone conversation). In contrast, ‘multicast’ services interconnect multiple nodes in the



**Fig. 22.7** Clustering multicast services. **(a)** Bipartite graph representation. **(b)** Optimal solution with two bicliques

network (for example a video-conference meeting). Specific efficient protocols for multicast services have been developed, and it is expected that the demand for multicast communication services will grow rapidly in the near future.

If we would choose to handle the multicast services optimally on an individual level, we would have to design and maintain a specific network configuration for each individual service, while ensuring global optimality of the network usage and provided services. Such a fine-grained individual approach is far from practical, as argued in [9]. Instead, multicast services are usually aggregated in clusters, and each cluster is considered as one meta-multicast service. For example, if several customers share a large number of services, they can be grouped together and use a network configuration that provides the information of all joint services to all customers. As a consequence, unnecessary information is sent between those customers and services in the cluster for which no relationship is required.

In [9], it is proposed to cluster the multicast sessions in such a way that the amount of unnecessary information sent through the network is limited, where this amount is measured by counting the number of client/service pairs between which unnecessary information is sent. An example is given in Fig. 22.7. In this example, we are given a set of multicast services  $I = \{i_1, i_2, i_3, i_4\}$  and a set of clients  $J = \{j_1, j_2, j_3, j_4, j_5\}$ . In Fig. 22.7a, the services and clients are depicted in a bipartite graph in which an edge  $(i, j)$  represents that client  $j$  requires service  $i$ . For example, client  $j_1$  requires services  $i_1$  and  $i_2$ . Figure 22.7b represents the clustering of the services into two sets  $\{i_1, i_2\}$  and  $\{i_3, i_4\}$ . This solution implies that unnecessary information is sent between the pairs  $(i_1, j_4)$ ,  $(i_2, j_3)$ ,  $(i_4, j_2)$  and  $(i_4, j_3)$ , as indicated

by the dashed edges in the figure. The total ‘cost’ of this solution is therefore 4. If we assume that the maximum number of clusters is 2, this solution is optimal.

Faure et al. [9] also suggest other variants of the problem that consider more fine-grained measures for the information that is sent through the network. For example, one can take into account path lengths or capacity of certain connections. Some variants can be encoded as weighted versions of the problem. A hybrid semidefinite and constraint programming approach for such weighted biclique completion problems was studied by Gualandi and Malucelli [28].

### 22.6.1 Problem Description

The specific variant considered by Gualandi [27] is described more formally as follows [9]. Let  $G = (I, J, D)$  be an undirected bipartite graph where vertices  $I$  represent a set of (multicast) services, vertices  $J$  represent a set of clients, and edge set  $D \subset I \times J$  represents the required demands.

We define  $p$  clusters  $T_1, \dots, T_p$  as follows. A cluster  $T_k = (I_k, J_k, D_k)$  is a subgraph of  $G$  where  $I_k \subset I$ ,  $J_k \subset J$ , and  $D_k = (I_k \times J_k) \cap D$ . Let  $K = \{1, \dots, p\}$  denote the index set of the clusters. Since the problem will treat each cluster as a single multicast group, the cluster will in practice correspond to a biclique (a complete bipartite graph) on the vertex sets  $I_k$  and  $J_k$ . The cost of cluster  $T_k$  is  $|I_k| \cdot |J_k| - |D_k|$ , corresponding to the amount (number) of unnecessary information sent to the clients.

Given a number  $p \in \{1, \dots, |J| - 1\}$ , the *multicast partition problem* consists of finding  $p$  clusters  $T_1, \dots, T_p$  such that  $I_1, \dots, I_p$  induce a partition of  $I$ , and the total cost

$$\sum_{k=1}^p |I_k| \cdot |J_k| - |D_k|$$

is minimized. Given the nature of the problem, it is also referred to in [9] and [27] as a ‘biclique completion’ problem. Faure et al. [9] show that the problem is NP-hard, even for  $p = 2$ .

### 22.6.2 Constraint Programming Model

The constraint programming model employed by Gualandi [27] relies on a custom-made global constraint, that will be applied to characterize each cluster and the associated cost. Let  $G = (S, T, E)$  be an undirected bipartite graph, and let  $\bar{G} = (S, T, \bar{E})$  be the complement graph of  $G$ , i.e.,  $\bar{E} = (S \times T) \setminus E$ . We introduce two set variables  $X$  and  $Y$  with initial domains  $D(X) = [\emptyset, S]$ ,  $D(Y) = [\emptyset, T]$ , and  $0 \leq |X| \leq |S|$ ,  $0 \leq |Y| \leq |T|$ . Variable  $X$  represents a set of nodes on one ‘shore’

of the graph, and all neighbors of  $X$  will be accumulated in  $Y$ . Further, let  $c$  be an integer variable with  $0 \leq c \leq |\overline{E}|$ . It corresponds to the number of edges in the complement graph  $\overline{G}$  that are necessary to create a biclique from the subgraph induced by  $X$  and  $Y$ . The specific global constraint, called ‘one-shore induced quasi-biclique constraint’ takes the form

$$\text{osi-biclique}(X, Y, c, G)$$

and states that

$$Y = \cup_{i \in X} N(i), \text{ and}$$

$$c = |X| \cdot |Y| - |F|,$$

where  $F = (X \times Y) \cap E$ . Here  $N(i)$  denotes the set of neighbors of a vertex  $i$ , i.e.,  $N(i) = \{j \mid (i, j) \in E\}$ . Gualandi [27] describes and implements specific domain filtering algorithms for this constraint.

We next build the constraint programming model for the original problem on a graph  $G = (I, J, D)$  with multicast services  $I$ , clients  $J$ , and demands  $D$  as specified above. We first introduce set variables  $X_1, \dots, X_p$  and  $Y_1, \dots, Y_p$ , where a pair  $(X_k, Y_k)$  represents the vertices of a cluster  $T_k = (I_k, J_k, D_k)$ . We further introduce an integer variable  $c_k$  representing the cost of cluster  $T_k$ , and a variable  $z$  representing the objective function. The model then becomes

$$\begin{aligned} \min z &= \sum_{k=1}^p c_k \\ \text{s.t. } &\text{partition}(X_1, \dots, X_p, I), \\ &\text{osi-biclique}(X_k, Y_k, c_k, G), \forall k \in K, \\ &D(X_k) = [\emptyset, I], 0 \leq |X_k| \leq |I|, \quad \forall k \in K, \\ &D(Y_k) = [\emptyset, J], 0 \leq |Y_k| \leq |J|, \quad \forall k \in K, \\ &0 \leq c_k \leq |\overline{E}|, \quad \forall k \in K, \\ &0 \leq z \leq |\overline{E}|. \end{aligned}$$

Recall that the `partition` constraint was defined in Sect. 22.2.

### 22.6.3 Semidefinite Programming Model

The semidefinite relaxation for this problem proposed by Gualandi [27] is closely related to the semidefinite relaxations for the MAX CUT problem [23] and the MAX  $k$ -CUT problem [18]. Let us first describe the semidefinite relaxation for  $p = 2$  and then extend this formulation for general  $p$ .

For  $p = 2$ , the objective of the multicast partition problem can be interpreted as minimizing the number of edges in the complement graph induced by the clusters  $T_1 = (I_1, J_1, D_1)$  and  $T_2 = (I_2, J_2, D_2)$ . This is equivalent to maximizing the number of edges in the complement graph that are in the cut between the sets  $(I_1 \cup J_1) \cap J_2$  and  $(I_2 \cup J_1) \cap J_1$ . That is, edges connected to clients that belong to both  $J_1$  and  $J_2$  are not considered in the cut.

For each vertex  $i \in I$  we introduce a variable  $x_i \in \{-1, 1\}$  representing whether  $i$  belongs to  $I_1$  ( $x_i = 1$ ) or to  $I_2$  ( $x_i = -1$ ). If two vertices  $i, j \in I$  are in the same cluster, the product  $x_i x_j$  equals 1, otherwise the product is equal to  $-1$ . We further introduce a variable  $z_{ij} \in \{-1, 1\}$  for every edge  $(i, j) \in \overline{E}$  representing whether  $(i, j)$  belongs to the cut ( $z_{ij} = -1$ ) or not ( $z_{ij} = 1$ ). Since an edge  $(i, j)$  does not belong to a cut if there exists a vertex  $k \in N(j)$  such that  $i$  and  $k$  belong to the same cluster, or  $x_i x_k = 1$ , we have  $z_{ij} = \max_{k \in N(j)} \{x_i x_k\}$ . This relation can be linearized as  $z_{ij} \geq x_i x_k$ , for all  $k \in N(j)$ .

The multicast partition problem with  $p = 2$  can now be formulated as the following quadratic program:

$$\begin{aligned} z_{QP} &= \max \frac{1}{2} \sum_{(i,j) \in \overline{E}} (1 - z_{ij}) \\ \text{s.t. } z_{ij} &\geq x_i x_k, \quad \forall (i,j) \in \overline{E}, k \in N(j), \\ x_i &\in \{-1, 1\}, \quad \forall i \in I, \\ z_{ij} &\in \{-1, 1\}, \quad \forall (i,j) \in \overline{E}. \end{aligned}$$

Note that the optimal solution value to the multicast partition problem is equal to  $|\overline{E}| - z_{QP}$ .

We next associate a unit vector  $\mathbf{v}_i \in \mathbb{R}^{|I|+|J|}$  to each vertex  $i \in I \cup J$ , with the interpretation that  $i$  and  $j$  are in the same cluster if  $\mathbf{v}_i \cdot \mathbf{v}_j = 1$ . Let  $V$  be the matrix consisting of columns  $\mathbf{v}_i$  for all  $i \in I \cup J$ , and let  $Z = V^T V$ . The semidefinite relaxation thus becomes:

$$\begin{aligned} \max \frac{1}{2} \sum_{(i,j) \in \overline{E}} (1 - Z_{ij}) \\ \text{s.t. } Z_{ij} &\geq Z_{ik}, \quad \forall (i,j) \in \overline{E}, k \in N(j), \\ \text{diag}(Z) &= \mathbf{e}, \\ Z &\succeq 0. \end{aligned}$$

Here  $\text{diag}(Z)$  represents the vector formed by the diagonal elements of  $Z$ .

When the number of clusters  $p$  is more than 2, the model can be altered similar to the approach taken in [18] for MAX  $k$ -CUT. We let  $\mathbf{a}_1, \dots, \mathbf{a}_p$  be unit vectors in  $\mathbb{R}^{p-1}$  satisfying  $\mathbf{a}_i \cdot \mathbf{a}_j = -\frac{1}{p-1}$  for  $i, j \in \{1, \dots, p\}, i \neq j$ . These vectors represent the different clusters. For each vertex  $i \in I$  we introduce a vector  $\mathbf{x}_i$  taking its value in  $\{\mathbf{a}_1, \dots, \mathbf{a}_p\}$ . That is, if two vertices  $i$  and  $j$  are in the same cluster we have  $\mathbf{x}_i \cdot \mathbf{x}_j = 1$ , otherwise we have  $\mathbf{x}_i \cdot \mathbf{x}_j = -\frac{1}{p-1}$ . We then introduce a variable  $z_{ij} \in \{-\frac{1}{p-1}, 1\}$

for each edge  $(i, j) \in \overline{E}$ , representing whether  $(i, j)$  is in the cut ( $z_{ij} = -\frac{1}{p-1}$ ) or not ( $z_{ij} = 1$ ). Using these variables, the multicast partition problem for general  $p$  can be formulated as

$$\begin{aligned} & \max \frac{p-1}{p} \sum_{(i,j) \in \overline{E}} (1 - z_{ij}) \\ \text{s.t. } & z_{ij} \geq \mathbf{x}_i \cdot \mathbf{x}_k, \quad \forall (i, j) \in \overline{E}, k \in N(j), \\ & \mathbf{x}_i \in \{\mathbf{a}_1, \dots, \mathbf{a}_p\}, \quad \forall i \in I, \\ & z_{ij} \in \left\{-\frac{1}{p-1}, 1\right\}, \quad \forall (i, j) \in \overline{E}. \end{aligned}$$

We can apply the same labeling technique as for the case  $p = 2$  above to obtain the following semidefinite relaxation for general  $p$ :

$$\begin{aligned} z_{\text{SDP}} = & \max \frac{p-1}{p} \sum_{(i,j) \in \overline{E}} (1 - Z_{ij}) \\ \text{s.t. } & Z_{ij} \geq Z_{ik}, \quad \forall (i, j) \in \overline{E}, k \in N(j), \\ & \text{diag}(Z) = \mathbf{e}, \\ & Z_{ij} \geq -\frac{1}{p-1}, \quad \forall i, j \in I \cup J, i \neq j, \\ & Z \succeq 0. \end{aligned}$$

Observe that for  $p = 2$ , the vectors  $\mathbf{a}_1$  and  $\mathbf{a}_2$  are in fact scalars, i.e.,  $a_1 = -1, a_2 = 1$ , in which case the latter model coincides precisely with the earlier model for  $p = 2$ .

The value of the semidefinite relaxation can be applied as a lower bound for the multicast partition problem using the relation  $z^* \geq |\overline{E}| - \lfloor z_{\text{SDP}} \rfloor$ , where  $z^*$  represents the optimal objective value for the multicast partition problem.

#### 22.6.4 Evaluation of the Hybrid Approach

As stated above, Gualandi [27] applies the semidefinite relaxation as a lower bound on the objective, but also to guide the constraint programming search process. To this end, entries in an optimal solution matrix  $Z^*$  to the semidefinite relaxation are interpreted as the likelihood that two vertices belong to the same cluster. That is, if  $Z_{ij}^*$  is close to  $-\frac{1}{p-1}$ ,  $i$  and  $j$  are not likely to be in the same cluster, whereas they are if  $Z_{ij}^*$  is close to 1. The closer that  $Z_{ij}^*$  is to the midpoint  $\frac{p}{2(p-1)}$ , the more uncertain it is whether  $i$  and  $j$  belong to the same cluster.

The search heuristic first finds a pair of vertices  $(i, j)$  with  $i, j \in I$ , that are not yet assigned to any cluster, such that  $|Z_{ij}^*| - \frac{p}{2(p-1)}$  is maximized. It then finds a variable  $X_k$  that contains at least one of these variables as a possible element. If  $Z_{ij}^* > \frac{p}{2(p-1)}$  it will assign both  $i$  and  $j$  to  $X_k$  as branching decision. Otherwise, it will assign either  $i$  or  $j$  to  $X_k$ .

The overall hybrid approach has two variants. The first computes and exploits the semidefinite relaxation at each node in the search tree. The second variant only solves the semidefinite relaxation once at the root node, as in [66]. In the computational experiments reported by Gualandi [27], these two approaches are compared against two other exact methods. The first uses a linearized quadratic integer programming model similar to [9], which is solved with IBM/ILOG CPLEX. The second is a pure constraint programming model, solved with Gecode, without the use of the semidefinite relaxation.

The computational results provide three main insights. First, the hybrid semidefinite and constraint programming approach is competitive to the integer programming approach (and in several cases better). Furthermore, in terms of efficiency, applying the semidefinite relaxation at each node in the search tree does not pay off. Instead, it was found that applying the semidefinite relaxation only once at the root node was more efficient. This strategy also outperformed the pure constraint programming approach.

## 22.7 Conclusion and Future Directions

In this chapter, we have described how constraint programming can be applied to model and solve combinatorial optimization problems, and how semidefinite programming has been successfully integrated into constraint programming. Specifically, we have shown that semidefinite relaxations can be a viable alternative to linear programming relaxations, that are commonly applied within optimization constraints. One of the main benefits of semidefinite relaxations in this context appears to be the accuracy when the solution to the semidefinite relaxation is applied as a search heuristic.

From a constraint programming perspective, arguably the most important question to be addressed is the application of semidefinite relaxations to design cost-based filtering algorithms for the variable domains, in addition to strengthening the bound on the objective that is currently done. Even though additional theory may be developed for this purpose, it is likely that specific algorithms must be designed and engineered, e.g., taking advantage of incremental data-structures, to make such application worthwhile in practice.

In order to make semidefinite programming relaxations more accessible to the constraint programming community, it would be worthwhile to investigate how the modeling and solving capabilities of constraint programming systems can be extended to facilitate this. For example, there exist various techniques to automatically create a linear programming relaxation from a given constraint programming model, see, e.g., [33, 35, 55]. Several systems, including the constraint programming solver Xpress Kalis [10] and the modeling language Zinc [5] provide an interface to embed and automatically exploit linear programming relaxations. Having such functionality for semidefinite programming relaxations would be helpful, especially for designing hybrid methods.

In addition, it would be very useful to develop hybrid semidefinite programming and constraint programming approaches for more applications. Through studying more diverse applications, we not only have a chance of improving the state of the art in solving those, but we can also gain more insight in the theoretical and practical characteristics that make such approaches successful.

From a semidefinite programming perspective, it may be worthwhile to consider alternatives to the ILP-inspired branch-and-bound and branch-and-cut solving methods, especially because most semidefinite relaxations are still relatively expensive to compute. This chapter offers several options. For example, one may consider not solving a complete SDP relaxation at each search node, but rather at selected nodes of the search tree. Furthermore, alternative search strategies, as presented in this chapter, may be considered. And of course, the (cost-based) filtering algorithms may be applicable directly through variable fixing procedures inside an SDP-based solver.

Finally, another issue that deserves future investigations is the relationship between constraint programming, integer programming, and semidefinite programming when handling symmetry in combinatorial problems. There exists a vast literature on explicit symmetry breaking in constraint programming and integer programming, see Gent et al. [20] and Margot [47] for recent surveys. In contrast, certain types of symmetry breaking are implicitly accounted for in semidefinite relaxations; see for example Anjos and Vannelli [2]. These complementary approaches could perhaps be combined very effectively.

In conclusion, even though several of the developments described in this chapter are still at an early stage, there clearly is a great potential for hybrid solution methods combining semidefinite programming and constraint programming.

**Acknowledgements** I would like to thank Stefano Gualandi for helpful comments on an earlier draft of the chapter.

## References

1. Anjos, M.F.: An improved semidefinite programming relaxation for the satisfiability problem. *Mathematical Programming* **102**(3), 589–608 (2005)
2. Anjos, M.F., Vannelli, A.: Computing Globally Optimal Solutions for Single-Row Layout Problems Using Semidefinite Programming and Cutting Planes. *INFORMS Journal on Computing* **20**, 611–617 (2008)
3. Apt, K.R.: The essence of constraint propagation. *Theoretical Computer Science* **221**(1–2), 179–210 (1999)
4. Apt, K.R.: *Principles of Constraint Programming*. Cambridge University Press, Cambridge (2003)
5. Brand, S., Duck, G.J., Puchinger, J., Stuckey, P.J.: Flexible, Rule-Based Constraint Model Linearisation. In: Hudak, P., Warren, D.S. (eds.) *Practical Aspects of Declarative Languages, 10th International Symposium (PADL 2008)*, San Francisco, January 2008. Lecture Notes in Computer Science, vol. 4902, pp. 68–83. Springer, Heidelberg (2008)

6. Charikar, M., Makarychev, K., Makarychev, Y.: Near-optimal algorithms for maximum constraint satisfaction problems. *ACM Transactions on Algorithms* **5**(3), 32:1–32:14 (2009)
7. Cheriyan, J., Cunningham, W.H., Tunçel, L., Wang, Y.: A Linear Programming and Rounding Approach to MAX 2-SAT. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **26**, 395–414 (1996)
8. Dechter, R.: *Constraint Processing*. Morgan Kaufmann, San Francisco (2003)
9. Faure, N., Chrétienne, P., Gourdin, E., Sourd, F.: Bipartite completion problems for multicast network design. *Discrete Optimization* **4**, 360–377 (2007)
10. FICO Xpress-Kalis User guide, Fair Isaac Corporation (2009)
11. Fischetti, M., Lodi, A.: Local Branching. *Mathematical Programming* **98**(1–3), 23–47 (2003)
12. Fleischman, D., Poggi de Aragão, M.V.: Improved SDP Bounds on the Exact Solution of Unconstrained Binary Quadratic Programming. Paper presented at the 2010 Optimization Days, Montréal, 10–12 May 2010
13. Focacci, F., Lodi, A., Milano, M.: Cost-based domain filtering. In: Jaffar, J. (ed.) *Principles and Practice of Constraint Programming*, 5th International Conference, CP'99, Alexandria, October 1999. Lecture Notes in Computer Science, vol. 1713, pp. 189–203. Springer, Heidelberg (1999)
14. Focacci, F., Lodi, A., Milano, M.: Optimization-Oriented Global Constraints. *Constraints* **7**(3–4), 351–365 (2002)
15. Focacci, F., Lodi, A., Milano, M.: Embedding relaxations in global constraints for solving TSP and TSPTW. *Annals of Mathematics and Artificial Intelligence* **34**(4), 291–311 (2002)
16. Focacci, F., Lodi, A., Milano, M.: A hybrid exact algorithm for the TSPTW. *INFORMS Journal on Computing* **14**(4), 403–417 (2002)
17. Focacci, F., Lodi, A., Milano, M., Vigo, D.: Solving TSP through the integration of OR and CP techniques. *Electronic Notes in Discrete Mathematics* **1**, 13–25 (1999)
18. Frieze A., Jerrum, M.: Improved Approximation Algorithms for MAX  $k$ -CUT and MAX BISECTION. *Algorithmica* **18**(1), 67–81 (1997)
19. Garey, M.R., Johnson, D.S.: *Computers and Intractability*. Freeman, New York (1979)
20. Gent, I.P., Petrie, K.E., Puget, J.-F.: Symmetry in Constraint Programming. In: Rossi, F., van Beek, P., Walsh, T. (eds.) *Handbook of Constraint Programming*, pp. 329–376. Elsevier (2006)
21. Gervet, C.: Conjunto: Constraint Logic Programming with Finite Set Domains. In: Bruynooghe, M. (ed.) *Proceedings of the International Logic Programming Symposium (ILPS)*, pp. 339–358. MIT Press, Cambridge (1994)
22. Goemans, M.X., Williamson, D.P.: New  $\frac{3}{4}$ -Approximation Algorithms for the Maximum Satisfiability Problem. *SIAM Journal on Discrete Mathematics* **7**(4), 656–666 (1994)
23. Goemans, M.X., Williamson, D.P.: Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *Journal of the ACM* **42**(6), 1115–1145 (1995)
24. Gomes, C.P., van Hoeve, W.J., Leahu, L.: The Power of Semidefinite Programming Relaxations for MAX-SAT. In: Beck, J.C., Smith, B. (eds.) *Proceedings of the Third International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR)*, Ireland, May/June 2006. Lecture Notes in Computer Science, vol. 3990, pp. 104–118. Springer, Heidelberg (2006)
25. Grötschel, M., Lovász, L., Schrijver, A.: *Geometric Algorithms and Combinatorial Optimization*. Wiley (1988)
26. Gruber, G., Rendl, F.: Computational experience with stable set relaxations. *SIAM Journal on Optimization* **13**(4), 1014–1028 (2003)
27. Gualandi, S.:  $k$ -Clustering Minimum Bipartite Completion via a Hybrid CP and SDP Approach. In: van Hoeve, W.-J., Hooker, J.N. (eds.) *Proceedings of the 6th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR)*, Pittsburgh, May 2009. Lecture Notes in Computer Science, vol. 5547, pp. 87–101. Springer, Heidelberg (2009)

28. Gualandi, S., Malucelli, F.: Weighted Bipartite Completion via CP-SDP Randomized Rounding. In: Bonami, P., Liberti, L., Miller, A., Sartenaer, A. (eds.) Proceedings of the European Workshop on Mixed Integer Nonlinear Programming, Marseille (2010)
29. Guruswami, V., Raghavendra, P.: Constraint Satisfaction over a Non-Boolean Domain: Approximation Algorithms and Unique-Games Hardness. In: Goel, A., Jansen, K., Rolim, J., Rubinfeld, R. (eds.) Proceedings of the 11th International Workshop on Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (APPROX) Boston, August 2008. Lecture Notes in Computer Science, vol. 5171, pp. 77–90. Springer, Heidelberg (2008)
30. Halperin, E., Zwick, U.: Approximation Algorithms for MAX 4-SAT and Rounding Procedures for Semidefinite Programs. *Journal of Algorithms* **40**, 184–211 (2001)
31. Harvey, W.D., Ginsberg, M.L.: Limited Discrepancy Search. In Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI), pp. 607–615. Morgan Kaufmann (1995)
32. Helmberg, C.: Fixing Variables in Semidefinite Relaxations. *SIAM Journal on Matrix Analysis and Applications* **21**(3), 952–969 (2000)
33. Hooker, J.: Logic-Based Methods for Optimization - Combining Optimization and Constraint Satisfaction. Wiley (2000)
34. Hooker, J.N.: Operations Research Methods in Constraint Programming. In: Rossi, F., van Beek, P., Walsh, T. (eds.) *Handbook of Constraint Programming*, pp. 527–570. Elsevier (2006)
35. Hooker, J.N.: Integrated methods for optimization. Springer (2007)
36. Joy, S., Mitchell, J., Borchers, B.: A branch and cut algorithm for MAX-SAT and weighted MAX-SAT. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **35**, 519–536 (1997)
37. Karloff, H., Zwick, U.: A 7/8-approximation algorithm for MAX 3SAT? In: Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 406–415. IEEE Computer Society (1997)
38. Katriel, I., Sellmann, M., Upfal, E., Van Hentenryck, P.: Propagating Knapsack Constraints in Sublinear Time. In: Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI), pp. 231–236. AAAI Press (2007)
39. de Klerk, E., van Maaren, H.: On semidefinite programming relaxations of (2+p)-SAT. *Annals of Mathematics and Artificial Intelligence* **37**, 285–305 (2003)
40. de Klerk, E., Warners, J.P.: Semidefinite Programming Approaches for MAX-2-SAT and MAX-3-SAT: computational perspectives. In: Pardalos, P.M., Migdalas, A., Burkard, R.E. (eds.) *Combinatorial and Global Optimization*, pp. 161–176. World Scientific (2002)
41. de Klerk, E., van Maaren, H., Warners, J.P.: Relaxations of the Satisfiability Problem Using Semidefinite Programming. *Journal of Automated Reasoning* **24**, 37–65 (2000)
42. Lau, H.C.: A New Approach for Weighted Constraint Satisfaction. *Constraints* **7**, 151–165 (2002)
43. Laurent, M., Poljak, S., Rendl, F.: Connections between semidefinite relaxations of the max-cut and stable set problems. *Mathematical Programming* **77**, 225–246 (1997)
44. Laurent, M., Rendl, F.: Semidefinite Programming and Integer Programming. In: Aardal, K., Nemhauser, G., Weismantel, R. (eds.) *Discrete Optimization*, Handbooks in Operations Research and Management Science, pp. 393–514. Elsevier (2005) Also available as Technical Report PNA-R0210, CWI, Amsterdam.
45. Leahu, L., Gomes, C.P.: Quality of LP-based Approximations for Highly Combinatorial Problems. In: Wallace, M. (ed.) *Proceedings of the Tenth International Conference on Principles and Practice of Constraint Programming (CP 2004)*, Toronto, September/October 2004. Lecture Notes in Computer Science, vol. 3258, pp. 377–392. Springer (2004)
46. Lovász, L.: On the Shannon capacity of a graph. *IEEE Transactions on Information Theory* **25**, 1–7, (1979)
47. Margot, F.: Symmetry in Integer Linear Programming. In: Jünger, M., Liebling, T.M., Naddef, D., Nemhauser, G.L., Pulleyblank, W.R., Reinelt, G., Rinaldi, G., Wolsey, L.A. (eds.) *50 Years of Integer Programming 1958–2008*, pp. 647–686. Springer (2010)

48. Milano, M. (ed.): *Constraint and Integer Programming - Toward a Unified Methodology*, Operations Research/Computer Science Interfaces, vol. 27. Kluwer Academic Publishers (2003)
49. Milano, M., van Hoeve, W.J.: Reduced Cost-Based Ranking for Generating Promising Subproblems. In: Van Hentenryck, P. (ed.) *Proceedings of the Eighth International Conference on Principles and Practice of Constraint Programming (CP)*, Ithaca, September 2002. Lecture Notes in Computer Science, vol. 2470, pp. 1–16. Springer (2002)
50. Nemhauser, G.L., Wolsey, L.A.: *Integer and Combinatorial Optimization*. Wiley (1988)
51. Puget, J.F.: PECOS: a high level constraint programming language. In: *Proceedings of the Singapore International Conference on Intelligent Systems (SPICIS'92)*, Singapore, 28 September–1 October 1992
52. Raghavendra, P.: Optimal Algorithms and Inapproximability Results for Every CSP? In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, Victoria, May 2008. pp. 245–254. ACM (2008)
53. Raghavendra, P., Steurer, D.: Integrality Gaps for Strong SDP Relaxations of UNIQUE GAMES. In: *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, Atlanta, October 2009. pp. 575–585. IEEE Computer Society (2009)
54. Raghavendra, P., Steurer, D.: How to Round Any CSP. In: *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, Atlanta, October 2009. pp. 586–594. IEEE Computer Society (2009)
55. Refalo, P.: Linear Formulation of Constraint Programming Models and Hybrid Solvers. In: Dechter, R. (ed.) *Proceedings of the Sixth International Conference on Principles and Practice of Constraint Programming (CP)*, Singapore, September 2000. Lecture Notes in Computer Science, vol. 1894, pp. 369–383. Springer (2000)
56. Régis, J.-C.: A Filtering Algorithm for Constraints of Difference in CSPs. In: *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI)*, vol. 1, pp. 362–367. AAAI Press (1994)
57. Régis, J.-C.: Cost-Based Arc Consistency for Global Cardinality Constraints. *Constraints* **7**, 387–405 (2002)
58. Régis, J.-C.: Global Constraints and Filtering Algorithms. In: Milano, M. (ed.) *Constraint and Integer Programming - Toward a Unified Methodology*, Series Operations Research/Computer Science Interfaces, vol. 27, 88–136. Kluwer Academic Publishers (2004)
59. Régis, J.-C.: Modélisation et Contraintes Globales en Programmation par Contraintes. Habilitation thesis, University of Nice (2004)
60. Rossi, F., van Beek, P., Walsh, T. (eds.): *Handbook of Constraint Programming*. Elsevier (2006)
61. Sadler, A., Gervet, C.: Global Filtering for the Disjointness Constraint on Fixed Cardinality Sets. Technical Report TR-IC-PARC-04-02, IC-PARC, Imperial College (2004)
62. Sellmann, M., Gellermann, T., Wright, R.: Cost-Based Filtering for Shorter Path Constraints. *Constraints* **12**(2), 207–238 (2007)
63. van Beek, P.: Backtracking search algorithms. In: Rossi, F., van Beek, P., Walsh, T. (eds.) *Handbook of Constraint Programming*, pp. 85–134. Elsevier, 2006.
64. Van Hentenryck, P., Perron, L., Puget, J.-F.: Search and strategies in OPL. *ACM Transactions on Computational Logic* **1**(2), 285–320 (2000)
65. van Hoeve, W.-J.: A hybrid constraint programming and semidefinite programming approach for the stable set problem. In: Rossi, F. (ed.) *Proceedings of the Ninth International Conference on Principles and Practice of Constraint Programming (CP 2003)*, Kinsale, September 2003. Lecture Notes in Computer Science, vol. 2833, pp. 407–421. Springer (2003)
66. van Hoeve, W.-J.: Exploiting Semidefinite Relaxations in Constraint Programming. *Computers and Operations Research* **33**(10), 2787–2804 (2006)
67. van Hoeve, W.-J., Katriel, I.: Global Constraints. In: Rossi, F., van Beek, P., Walsh, T. (eds.) *Handbook of Constraint Programming*, pp. 169–208. Elsevier, 2006.
68. van Hoeve, W.-J., Pesant, G., Rousseau, L.-M.: On global warming: Flow-based soft global constraints. *Journal of Heuristics* **12**(4), 347–373 (2006)

69. Warners, J.: Nonlinear Approaches to Satisfiability Problems. Ph.D. thesis, Technische Universiteit Eindhoven (1999)
70. Xing, Z., Zhang, W.: MaxSolver: An efficient exact algorithm for (weighted) maximum satisfiability. *Artificial Intelligence* **164**(1–2), 47–80 (2005)
71. Zwick, U.: Approximation Algorithms for Constraint Satisfaction Problems Involving at Most Three Variables per Constraint. In: Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), San Francisco, 25–27 January 1998

## **Part III**

# **Software**

# Chapter 23

## The State-of-the-Art in Conic Optimization Software

Hans D. Mittelmann

### 23.1 Introduction

The following two sections are taken from [18]. In the first section the optimization problems under consideration are defined and the needed notation is introduced. In the second section a comprehensive set of error measures is given which helps evaluate and compare different codes. These measures have become standard since the 7th DIMACS Challenge and are produced by most of the codes listed in the following.

#### 23.1.1 The Problems Solved

The primal and dual pair of cone optimization problems over a self-dual cone are defined as

$$(P) \quad \begin{array}{ll} \min & \langle c, x \rangle \\ \text{s.t.} & x \in K \\ & \mathcal{A}x = b \end{array} \quad (D) \quad \begin{array}{ll} \max & b^T y \\ \text{s.t.} & z \in K \\ & \mathcal{A}^*y + z = c \end{array}$$

where

- $K$  is a closed, convex cone in a Euclidean space  $X$ .
- $\mathcal{A}: X \rightarrow \mathbb{R}^m$  is a linear operator, and  $\mathcal{A}^*$  is its adjoint.
- $b \in \mathbb{R}^m$ , and  $c \in X$ .

---

H.D. Mittelmann (✉)  
School of Mathematical and Statistical Sciences, Arizona State University,  
Tempe, AZ 85287, USA  
e-mail: [mittelmann@asu.edu](mailto:mittelmann@asu.edu)

In the case of a semidefinite-quadratic-linear program these are defined as follows:

- *The space X*:  $x \in X \Leftrightarrow x = (x_1^s, \dots, x_{n_s}^s, x_1^q, \dots, x_{n_q}^q, x^\ell)$ , where
  - $x_1^s, \dots, x_{n_s}^s$  are symmetric matrices (possibly of various sizes).
  - $x_1^q, \dots, x_{n_q}^q$  are vectors (again, possibly of various sizes).
  - $x^\ell$  is a vector.
- *The cone K*:  $x \in K \Leftrightarrow x_j^s \geq 0 \forall j, x_k^q \geq_q 0 \forall k, x^\ell \geq 0$ , where
  - $u \geq 0$  means that the symmetric matrix  $u$  is positive semidefinite.
  - $v \geq_q 0$  means that the vector  $v$  is in a quadratic cone (also known as the second-order cone, Lorentz cone or ice cream cone) of appropriate size. That is, if  $v \in \mathbb{R}^k$ , then  $v \geq_q 0 \Leftrightarrow v_1 \geq \|v_{2:k}\|_2$ .
  - $w \geq 0$  means that the vector  $w$  is componentwise nonnegative.
- *The inner product  $\langle \cdot, \cdot \rangle$* : For  $x, z \in K$ 

$$\langle x, z \rangle = \sum_{j=1}^{n_s} x_j^s \bullet z_j^s + \sum_{k=1}^{n_q} x_k^q * z_k^q + x^\ell * z^\ell,$$

where

- For matrices  $a$  and  $b$ ,  $a \bullet b = \text{trace } a^T b = \sum_{i,j} a_{ij} b_{ij}$ .
- For vectors  $a$  and  $b$ ,  $a * b = a^T b = \sum_i a_i b_i$ .

In the following  $a_{ij}$  and  $a_i$  denote the respective matrix and vector parts of the operators  $\mathcal{A}$  and  $\mathcal{A}^*$ . Thus (P) and (D) become

$$\begin{aligned} \min & \sum_{j=1}^{n_s} c_j^s \bullet x_j^s + \sum_{k=1}^{n_q} c_k^q * x_k^q + c^\ell * x^\ell \\ \text{s.t.} & \sum_{j=1}^{n_s} a_{ij}^s \bullet x_j^s + \sum_{k=1}^{n_q} a_{ik}^q * x_k^q + a_i^\ell * x^\ell = b_i \quad \forall i \\ & x_j^s \geq 0 \quad \forall j \quad x_k^q \geq_q 0 \quad \forall k \quad x^\ell \geq 0 \end{aligned} \tag{S QLP - P}$$

and

$$\begin{aligned} \max & \sum_{i=1}^m b_i y_i \\ \text{s.t.} & \sum_{i=1}^m a_{ij}^s y_i + z_j^s = c_j^s \quad \forall j \\ & \sum_{i=1}^m a_{ij}^q y_i + z_k^q = c_j^q \quad \forall k \\ & \sum_{i=1}^m a_i^\ell y_i + z^\ell = c^\ell \\ & z_j^s \geq 0 \quad z_k^q \geq_q 0 \quad z^\ell \geq 0 \end{aligned} \tag{S QLP - D}$$

Thus the feasible set is a product of semidefinite, quadratic and nonnegative orthant cones, intersected with an affine subspace. It is possible that one or more of the three parts of the problem is absent, i.e., any of  $n_s$ ,  $n_q$ , or the length of  $x^\ell$  may be zero.

The *rotated quadratic cones* is defined as

$$\left\{ v \in \mathbb{R}^k \mid v_1 v_2 \geq \|v_{3:k}\| \right\}.$$

It is simply a rotation of the usual quadratic cone, but for the purpose of modeling quadratic inequalities, it is often more convenient to use, thus several participating codes support this cone.

### 23.1.2 Computing Errors

Suppose that we are given *approximate* optimal solutions of  $(SQLP - P)$  and  $(SQLP - D)$ , respectively. To compute how far they are from an exact solution pair, we define a norm on  $X$ , and a minimum eigenvalue with respect to the cone  $K$ . Precisely, if  $x \in X$ , then

$$\|x\| = \sum_{j=1}^{n_s} |x_j^s|_F + \sum_{k=1}^{n_q} |x_k^q|_2 + |x^\ell|_2,$$

where for a matrix  $a$ ,  $\|a\|_F$  is the Frobenius norm of  $a$ . Also,

$$\lambda_{\min,K}(x) = \min \left\{ \min_{j=1,\dots,n_s} \lambda_{\min}(x_j^s), \min_{k=1,\dots,n_q} \lambda_{\min,q}(x_k^q), \min_h x_h^\ell \right\}$$

where

- For a symmetric matrix  $a$ ,  $\lambda_{\min}(a)$  is the usual smallest eigenvalue of  $a$ .
- For a vector  $a$ ,  $\lambda_{\min,q}(a) = a_1 - |a_{2:k}|_2$ .

Also, we denote by  $|a|_1$  the absolute value of the largest component of  $a$ , for an arbitrary  $a \in X$ .

Then, for approximate optimal solutions  $x$  of  $(SQLP - P)$  and  $(y, z)$  of  $(SQLP - D)$ , we define

$$\begin{aligned} \text{err}_1(x, y, z) &= \frac{\|\mathcal{A}x - b\|_2}{1 + \|b\|_1} \\ \text{err}_2(x, y, z) &= \max \left\{ 0, \frac{-\lambda_{\min,K}(x)}{1 + \|b\|_1} \right\} \\ \text{err}_3(x, y, z) &= \frac{\|\mathcal{A}^*y + z - c\|_2}{1 + \|c\|_1} \end{aligned}$$

$$\text{err}_4(x, y, z) = \max\left\{0, \frac{-\lambda_{\min, K}(z)}{1 + |c|_1}\right\}$$

$$\text{err}_5(x, y, z) = \frac{\langle c, x \rangle - b^T y}{1 + |\langle c, x \rangle| + |b^T y|}$$

Furthermore, when  $x$  and  $z$  are both in  $K$ , that is  $\text{err}_2(x, y, z) = \text{err}_4(x, y, z) = 0$ , we also define

$$\text{err}_6(x, y, z) = \frac{\langle x, z \rangle}{1 + |\langle c, x \rangle| + |b^T y|}$$

A few remarks are in order.

- If  $x$  and  $z$  are both feasible, then in exact arithmetic  $\text{err}_5(x, y, z) = \text{err}_6(x, y, z)$ .
- As  $x$  and  $z$  are *approximate* optimal solutions only, we may have  $\text{err}_5(x, y, z) < 0$ . It is possible that, all other error measures being the same, if  $\text{err}_5(x, y, z) = -\delta$  with  $\delta > 0$ , then  $(x, y, z)$  corresponds to a solution that is “worse”, than if  $\text{err}_5(x, y, z)$  were  $\delta$ . Thus, we decided to report  $\text{err}_5(x, y, z)$ , not merely the maximum of  $\text{err}_5(x, y, z)$  and 0 (as done in several papers), so as not to suppress any information.
- Several codes do not explicitly maintain  $z$ ; in this case, one should set

$$z = c - \mathcal{A}^* y$$

Of course, then  $\text{err}_3(x, y, z)$  will be zero (depending on the accuracy achieved by the computer).

The above six error measures were introduced in [18]. They are evaluated by most conic codes and their output, in addition to any solver-specific measures, is routine or may be activated (SDPA).

The evaluation of conic optimization software was continued after [18] in the webpage [20]. This comparison reflects in mid-2010 the status of mid-2008 with a few exceptions. The reason is that the software considered already in [20] has practically not undergone any major algorithmic development. Nevertheless, it is planned to update the conic part of [20] within the next 12 months when some major developments are expected, notably the rewriting of DSDP, the future development of SMCP, and extensions such as that of MOSEK to SDP. One major code improvement that has taken place lately is documented separately, namely the parallelization and extension to multiple precision of SDPA [11].

In the following we devote one paragraph to each of the codes similar to [18]. These have partly been coordinated with the authors. A code not included but part of the benchmarks [20] is PENSDP. It is one of the very few codes able to solve nonlinear SDP problems. A separate article in this volume is devoted to the code PENNON which PENSDP is a part of.

## 23.2 The Codes

The submitted codes fall naturally into two major groups and several subgroups. For links to this and related software see also [19].

- The first major group is that of primal–dual interior point methods designed for small to medium sized problems.
  - In the first subgroup are the codes SeDuMi, SDPT3, and SMCP. These codes handle all three types of cones.
  - In the second subgroup are SDPA and CSDP, which are limited to SDP.
  - In the third subgroup are codes that had started out as LP codes, CPLEX, LOQO, and MOSEK. They were extended to QP, SOCP, and convex (MOSEK) or nonconvex (LOQO) nonlinear optimization.
- The second group is that of large-scale SDP codes designed to provide approximate solutions for large scale problems: SDPLR, ConicBundle, and DSDP. The first two of these do not make use of second order derivative information, while DSDP is a dual interior point code.

The major input formats are:

**Matlab:** SeDuMi format in Matlab binary form [24]

**QPS:** extended MPS format, different for MOSEK and CPLEX

**SDPA:** the sparse SDPA format as explained in the SDPA user's guide

**Language:** Codes that have no specific input format (ConicBundle); other codes can be used from one or more languages as indicated

### 23.2.1 SDPA

**Authors:** K. Fujisawa, M. Fukuda, Y. Futakata, K. Kobayashi, M. Kojima, K. Nakata, M. Nakata, and M. Yamashita

**Version** 7.3.1, 7/2009

**Available:** yes; at <http://sdpa.indsys.chuo-u.ac.jp/sdpa/>

**Key paper:** [11]. See [27] for implementation and numerical results.

**Features:** primal–dual method

**Language; Input format:** C++; SDPA, Matlab, SeDuMi

**Error computations:** yes

**Solves:** SDP

SDPA (SemiDefinite Programming Algorithm) is based on a Mehrotra-type predictor–corrector infeasible primal–dual interior-point method with the H.K.M direction. It is implemented in C++ language utilizing BLAS/LAPACK libraries for dense matrix computations and MUMPS for sparse Cholesky factorizations. SDPA exploits sparsity of data input matrices and employs multi-threading computing

to solve semidefinite programs in practice in a short time. SDPA is available as a callable library, too. In addition, it supports a Matlab interface and accepts SDPA or SeDuMi input formats.

SDPA uses a set of parameters which the user can adjust to cope with numerically difficult semidefinite programs. Stopping criteria:

$$\begin{aligned} & - \min \{ \| \mathcal{A}x - b \|_{\infty}, \| \mathcal{A}^*y + z - c \|_{\infty} \} \leq \epsilon' \quad \text{and} \\ & - \frac{|\langle c, x \rangle - b^T y|}{\max \{ |\langle c, x \rangle| + |b^T y|, 2.0, 1.0 \}} \leq \epsilon^* \end{aligned}$$

Typical values of the parameters  $\epsilon'$  and  $\epsilon^*$  are from  $10^{-8}$  to  $10^{-6}$ .

**Remarks:** In addition, SDPA has some variants which all together are named SDPA Family. SDPA-GMP is an implementation of the algorithms with arbitrary accuracy libraries and enables to attain  $\epsilon' = \epsilon^* = 10^{-30}$ , for example. A parallel version, SDPARA, solves extremely large-scale semidefinite programs on multiple PCs connected by high-speed networks.

The contribution by the SDPA authors in this volume will introduce each software of the SDPA Family.

### 23.2.2 *SeDuMi*

**Authors:** J. Sturm, I. Polik

**Version:** 1.3, 4/2010

**Available:** yes; from <http://sedumi.ie.lehigh.edu/>

**Key papers:** [24, 25]

**Features:** self-dual embedding, dense column handling

**Language; Input format:** Matlab+C; Matlab, SDPA

**Error computations:** yes

**Solves:** SDP, SOCP

The primal–dual interior point algorithm implemented in SeDuMi [24] is described in [25]. The algorithm has an  $O(\sqrt{n} \log \epsilon)$  worst case bound, and treats initialization issues by means of the self-dual embedding technique of [28]. The iterative solutions are updated in a product form, which makes it possible to provide highly accurate solutions.

The algorithm terminates successfully if the norm of the residuals in the optimality conditions, or the Farkas system with  $b^T y = 1$  or  $\langle c, x \rangle = -1$ , are less than the parameter `pars.eps`. The default value for `pars.eps` is `1E-9`.

**Remark:**

- SeDuMi exploits sparsity in solving the normal equations; this results in a benefit for problems with a large number of small order matrix variables, such as the *copositivity*-problems in the DIMACS set.

- However, for problems that involve a huge matrix variable (without a block diagonal structure), the implementation is slow and consumes an excessive amount of memory.

### 23.2.3 CSDP

**Author:** B. Borchers

**Version:** 6.1.1, 4/2010

**Available:** yes; from <http://projects.coin-or.org/Csdp/>

**Key paper:** [6]

**Features:** infeasible predictor–corrector variant of a primal dual method based on the H..K..M direction

**Language; Input format:** C; SDPA, Interfaces to Matlab/Octave and R

**Error computations:** yes

**Solves:** SDP

CSDP consists of a callable library and standalone solver for SDP problems. It is not applicable to problems with second order cone constraints. The code uses a predictor–corrector variant of the primal–dual method of Helmberg et al. [14] and Kojima et al. [17]. CSDP is suited to the solution of small and medium size SDPs with general structure. The algorithm supports matrices with block diagonal structure as well as linear programming variables which are expressed as a diagonal block within the SDP problem.

CSDP is written in portable ANSI C with calls to subroutines from either the Linpack or LAPACK libraries. The required Linpack routines are supplied with the code. However, improved performance can be obtained by using BLAS and LAPACK libraries that have been optimized for a particular computer.

The default stopping criteria are:

- $\frac{|\langle c, \bar{x} \rangle - b^T \bar{y}|}{1 + |b^T \bar{y}|} < 10^{-7}$  and
- $\frac{\|\mathcal{A}\bar{x} - b\|_2}{1 + \|b\|_2} < 10^{-8}$  and
- $\frac{\|\mathcal{A}^* \bar{y} - c + z\|_2}{1 + \|c\|_2} < 10^{-8}$

In addition, CSDP maintains positive definite  $X$  and  $Z$  matrices at all times. CSDP checks this by computing the Cholesky factorizations of  $X$  and  $Z$ .

CSDP uses OPENMP to parallelize on shared memory multiprocessor systems; see [7].

### 23.2.4 DSDP

**Authors:** S. Benson, Y. Ye

**Version:** 5.8, 10/2005

**Available:** yes; from <http://www.mcs.anl.gov/hs/software/DSDP/>

**Key paper:** [5]

**Features:** Dual scaling potential reduction method, rank-1 constraints, Matlab interface, MPI parallel

**Language; Input format:** C; SDPA

**Error computations:** yes

**Solves:** SDP

The DSDP software package is an implementation of the dual scaling algorithm for SDP. Unlike primal–dual interior point methods, this algorithm uses only the dual solution to generate a step direction. It can generate feasible primal solutions as well, but it saves time and memory if this feature is not used. Many large problems have well structured constraints in the form of low rank matrices or sparse matrices. DSDP explicitly accounts for these features by using sparse data structures for the dual matrix and the eigenvalue/eigenvector decomposition of the data matrices. Theoretical convergence results exist for feasible starting points, and strong computational results have been achieved using feasible and infeasible starting points. DSDP initially solves the Schur complement equations using the conjugate residual method, and then it switches to the Cholesky method when the conditioning of the matrix worsens.

Stopping criteria:

$$|\langle c, x \rangle - b^T y| / (|b^T y| + 1) \leq 10^{-3} \quad \text{and}$$

$$\|\mathcal{A}^* y + z - c\|_\infty \leq 10^{-8}$$

or

$$|b^T y| \geq 10^8 \quad (\text{unbounded dual})$$

or

stepsizes less than  $10^{-3}$  for more than 10 iterations (dual infeasibility)

In the summer of 2010 a new version will be released based on a hybrid of Quasi-Newton and Newton methods with a Hessian-Update technique.

### 23.2.5 SDPT3

**Authors:** K.C. Toh, M. Todd, R. Tütüncü

**Version:** 4.0, 2/2009

**Available:** yes; from <http://www.math.nus.edu.sg/~mattohkc/sdpt3.html>

**Key paper:** [26]

**Features:** primal–dual method, infeasible primal–dual and homogeneous self-dual formulations, Lanczos steplength computation

**Language; Input format:** Matlab+C or Fortran; SDPA

**Error computations:** yes

**Solves:** SDP, SOCP

This code is designed to solve conic programming problems whose constraint cone is a product of semidefinite cones, second-order cones, and/or nonnegative orthants. It employs a predictor–corrector primal–dual path-following method, with either the H.K.M or the NT search direction. The basic code is written in Matlab, but key subroutines in Fortran and C are incorporated via Mex files. Routines are provided to read in problems in either SeDuMi or SDPA format. Sparsity and block diagonal structure are exploited, but the latter needs to be given explicitly.

The algorithm is stopped if:

- Primal infeasibility is suggested because  $b^T y / \|\mathcal{A}^* y + z\| > 10^8$ ; or
- Dual infeasibility is suggested because  $-\langle c, x \rangle / \|\mathcal{A}x\| > 10^8$ ; or
- Sufficiently accurate solutions have been obtained:

$$\text{rel\_gap} := \frac{\langle x, z \rangle}{\max\{1, (\langle c, x \rangle + b^T y) / 2\}}$$

and

$$\text{infeas\_meas} := \max \left[ \frac{\|\mathcal{A}x - b\|}{\max\{1, \|b\|\}}, \frac{\|\mathcal{A}^* y + z - c\|}{\max\{1, \|c\|\}} \right]$$

are both below  $10^{-8}$ ; or

- Slow progress is detected, measured by a rather complicated set of tests including

$$\langle x, z \rangle / n < 10^{-4} \quad \text{and} \quad \text{rel\_gap} < 5 * \text{infeas\_meas};$$

or

- Numerical problems are encountered, such as the iterates not being positive definite, the Schur complement matrix not being positive definite, or the step sizes falling below  $10^{-6}$ .

### Remark

- SDPT3 is a general-purpose code based on a polynomial-time interior-point method.
- It should obtain reasonably accurate solutions to problems of small and medium size (for problems with semidefinite constraints, up to around a thousand constraints involving matrices of order up to around a thousand, and for sparse problems with only second-order/linear cones, up to around 20,000 constraints and 50,000 variables), and can solve some larger problems.
- Because it uses a primal–dual strategy, forms the Schur complement matrix for the Newton equations, and employs direct methods, it is unlikely to compete favorably with alternative methods on large-scale problems.

### New Features:

- Free variables
- Determinant maximization problems
- SDP with complex data
- Three-parameter homogeneous self-dual model of SQLP

### 23.2.6 MOSEK

**Author:** E. Andersen

**Version:** 6.0, 7/2010

**Available:** yes; from <http://www.mosek.com/>

**Key paper:** [2]

**Features:** Solves large scale sparse SOCPs. Is parallelized. Solves mixed-integer SOCPs

**Interfaces:** C, Java, .NET, MATLAB, Python

**Modeling Language interfaces:** GAMS, AIMMS, AMPL

**Input formats:** OPF, Extended MPS

**Error computations:** yes

**Solves:** SOCP

The conic quadratic optimizer implemented in MOSEK employs the NT search direction. The other main features of the implementation are that it is based on a homogeneous and self-dual model, handles the rotated quadratic cone directly, employs a Mehrotra type predictor–corrector extension and sparse linear algebra to improve the computational efficiency.

Other features:

- Has an extensive presolve facility to reduce problem size before optimizing
- Exploits special structure in rotated quadratic cones
- Exploits fixed and upper bounded variables
- Detects primal and dual infeasibility reliably

Future plans: Support for SDPs.

### 23.2.7 LOQO

**Authors:** H.Y. Benson, R. Vanderbei

**Version:** 6.07, 9/2006

**Available:** yes; from <http://www.princeton.edu/~rvdb/>

**Key paper:** [4]

**Features:** NLP approach

**Language; Input format:** C; SDPA, Matlab, AMPL

**Error computations:** no

**Solves:** SOCP

LOQO is a software package for solving general (smooth) nonlinear optimization problems of the form

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && g_i(x) = 0, \quad i \in \mathcal{E} \\ & && h_i(x) \geq 0, \quad i \in \mathcal{I}, \end{aligned}$$

where  $x \in \mathbb{R}^n$ ,  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ ,  $\mathcal{E}$  is the set of equalities,  $\mathcal{I}$  is the set of inequalities,  $g : \mathbb{R}^n \rightarrow \mathbb{R}^{|\mathcal{E}|}$ , and  $h : \mathbb{R}^n \rightarrow \mathbb{R}^{|\mathcal{I}|}$ . It implements an infeasible-primal-dual path-following method and requires that the problem be smooth, that is  $f$  and  $h$  be twice differentiable, and  $g$  be an affine function. Even though LOQO can handle nonconvex problems in general, it performs better with convex problems, where  $f$  is convex,  $g$  are affine, and  $h$  are concave functions.

Stopping criteria

$$\|\mathcal{A}x - b\|_2 \leq 10^{-7}$$

$$\|\mathcal{A}^*y + z - c\|_2 \leq 10^{-8}$$

$$\log_{10} \left( \frac{|\langle c, x \rangle - b^T y|}{|\langle c, x \rangle| + 1} \right) \leq -8$$

- LOQO can handle other types of nonlinear constraints in the problem.
- A weakness results from the use of the Cholesky factorization. LOQO works best with a sparse problem, and it does not exploit the special structure of an SOCP.
- Its SDP approach leads to dense problems.

### 23.2.8 SDPLR

**Authors:** S. Burer, R. Monteiro

**Version:** 1.03beta, 6/2009

**Available:** yes; from <http://dollar.biz.uiowa.edu/~sburer/doku.php>

**Key paper:** [8]

**Features:** Augmented Lagrangian algorithm applied to an NLP formulation of an SDP arising by replacing the primal variable by a low-rank factorization.

**Language; Input format:** C; SDPA, SDPLR

**Error computations:** yes

**Solves:** SDP

The code SDPLR is an infeasible, nonlinear programming method for solving any standard form primal SDP.

The main idea of SDPLR is to eliminate the primal positive semidefiniteness constraint  $X \succeq 0$  by employing an implicit factorization  $X = RR^T$  that is valid for

at least one optimal solution (but not necessarily for all feasible solutions). Using a theorem of Pataki,  $R$  is chosen to be an  $n \times r$  matrix, where  $r$  is the smallest integer satisfying  $r(r+1)/2 \geq m$  and  $m$  is the number of primal constraints (aside from the constraint  $X \succeq 0$ ). A stationary point  $\bar{R}$  of the new nonlinear problem is then found using a first-order augmented Lagrangian method, and in practice, such a stationary point reliably gives an optimal SDP solution  $\bar{X} = \bar{R}\bar{R}^T$ .

SDPLR is an infeasible method that is terminated once the norm of the primal infeasibility has been reduced to a value of  $10^{-6}$ , which, in accordance with the augmented Lagrangian algorithm, indicates an approximate stationary point.

- The main strength of SDPLR is that it is a first-order algorithm and hence can attack large-scale SDPs.
- The infeasible nature of SDPLR is a strong disadvantage.
- SDPLR does have a convergence theory that explains its practical convergence, though the theory is based on some assumptions of sequences, etc, and is not as strong as, for example, the theory of regular interior-point methods. The theory is contained in [9].
- SDPLR works with density of the dual matrix  $S$  and moreover works with a small number of columns  $r$ . As a result, the iterations of SDPLR are extremely fast.

### 23.2.9 ConicBundle

**Authors:** C. Helmberg

**Version:** 0.3.5, 4/2010

**Available:** yes; from <http://www.tu-chemnitz.de/~helmberg/ConicBundle>

**Key papers:** [3, 13, 15, 16]

**Features:** nonsmooth convex optimization with special purpose models for Second Order Conic and Semidefinite Optimization and additional support for Lagrangian relaxation/decomposition

**Language; Input format:** C++; C++

**Error computations:** no; only norm of an aggregate subgradient

**Solves:** Convex optimization problems given by first order oracles.

Special oracles and models are provided for eigenvalue optimization problems of the form

$$\min_{y \in \mathbb{R}^m} a\lambda_{\max} \left( C - \sum_{i=1}^m A_i y_i \right) + b^T y$$

The design variables  $y_i$  may be box constrained,  $C$  and the  $A_i$  are given real symmetric matrices,  $b \in \mathbb{R}^m$  allows to specify a linear cost term, and  $a < 0$  is a constant multiplier for the maximum eigenvalue function  $\lambda_{\max}(\cdot)$ . The code is intended for large scale problems and allows to exploit structural properties of the matrices such as sparsity and low rank structure. From a Lagrangian relaxation perspective, the code solves the dual to programs of the form  $\max\{\langle C, X \rangle : X \succeq 0, \langle A_i, X \rangle = b_i, i =$

$1, \dots, m\}$  with bounded trace,  $\text{tr}X \leq a$ , provides support for several positive matrix variables as well as for adding cutting planes and accumulates solution information to form a sparse or primal aggregate  $X^*$  that may be used as an approximation to an optimal  $X$ . The code is a dynamic bundle method constructing a minorizing model of the Lagrangian dual based on subgradient information in the style of the proximal bundle method of [17]. Function values and subgradients of the nonsmooth convex function  $f(y) := a\lambda_{\max}(C - \sum_{i=1}^m A_i y_i) + b^T y$  are determined via computing the maximum eigenvalue and a corresponding eigenvector by a Lanczos method. The code never factorizes the matrix  $C - \sum_{i=1}^m A_i y_i$  but uses matrix vector multiplications exclusively. Thus, there is no danger of increasing memory requirements or computational work by additional fill-in. Experimentally, the method seems to exhibit fast convergence if the semidefinite subcone that is used to generate the semidefinite cutting surface model, spans the optimal solution of the corresponding primal problem. If the subcone, however, is too small, the typical tailing off effect of subgradient methods is observed once a relative precision of roughly  $10^{-3}$  has been reached.

### 23.2.10 SMCP

**Authors:** E. Andersen, L. Vandenberghe

**Version:** 0.3a, 5/2010

**Available:** yes; from <http://abel.ee.ucla.edu/smcp/>

**Key papers:** [1]

**Features:** primal and dual scaling methods for sparse matrix conic programs

**Language; Input format:** C; Python, SDPA or CVXOPT matrix

**Error computations:** yes

**Solves:** SDP

SMCP implements feasible start primal and dual barrier methods for sparse matrix conic programs (which include SDP, SOCP, and LP as special cases). The algorithm takes advantage of chordal sparsity when solving the Newton equations. It is implemented in Python/C and relies on the Python extensions CVXOPT and CHOMPACK for linear algebra and chordal matrix computations.

Stopping criteria:

$$\frac{\|b - \mathcal{A}x\|_2}{\max\{1, \|b\|_2\}} \leq \epsilon_{\text{feas}}, \quad \frac{\|\mathcal{A}^{\text{adj}}y + z - z\|_F}{\max\{1, \|c\|_F\}} \leq \epsilon_{\text{feas}}, \quad x \succ_K^* 0, \quad z \succ_K$$

and

$$\langle x, z \rangle \leq \epsilon_{\text{abs}}, \quad \left( \min\{\langle c, x, -b^T y \rangle\} \leq 0, \frac{\langle x, z \rangle}{-\min\{\langle c, x, -b^T y \rangle\}} \right) \leq \epsilon_{\text{rel}}$$

where  $\epsilon_{\text{feas}} = 10^{-8}$  and  $\epsilon_{\text{rel}} = \epsilon_{\text{abs}} = 10^{-6}$  are the default tolerances.

### 23.2.11 CPLEX

**Authors:** IBM

**Version:** 12.2, 6/2010

**Available:** yes; from <http://www.ibm.com/academiclicense>)

**Key papers:** see documentation

**Features:** specialized SOCP solver

**Language:** C, C++, C#, VB.net, Java, Matlab, Microsoft Excel, Python

**Input format:** extended MPS

**Error computations:** yes

**Solves:** standard and rotated SOCP versions

In addition to linear and quadratic programs, and their mixed integer versions, IBM ILOG CPLEX also solves programs involving standard and rotated cones, and convex quadratic constraints, both the continuous and the mixed integer versions.

Rotated cones and convex quadratic constraints are handled by transforming them into standard conic form. The standard conic form is then solved by a specialized SOCP solver. This solver is based on the homogeneous self dual formulation and Nesterov–Todd scaling. The formulation is solved in scaled space. A Cholesky factorization is used to solve the normal system and dense columns are handled in product form. The solver uses the predictor corrector technique.

The code is written C, and uses very efficient techniques for performing sparse linear algebra.

The stopping criterion is based on the primal, dual and relative gap error. It is controlled by the `qpconvergence` tolerance parameter which is  $10^{-7}$  by default. The quality of the solutions can be displayed using the IBM ILOG CPLEX solution quality queries.

## 23.3 Comments

While no tables with comparisons as in [18] will be given here, a few remarks are in order for a potential user of one of the codes. Based on the current status for SDP problems CSDP, SeDuMi, SDPA, and SDPT3 are all quite stable. SMCP will be extended from its current feasible version but is partly already competitive as shown in [1]. We do not comment here on DSDP since a rewrite of it is imminent. Parallelization, especially of SDPA for both distributed and, now becoming very important, for shared memory, leads to a significant speed up. Also, the many ill-conditioned SDPs are currently solved best with the multiple precision versions of SDPA. They have been installed by us for use through [22] and also have been instrumental in our recent research [12,21].

For problems of special structure or more general than SDP/SOCP, the codes SDPLR, PENNON and ConicBundle are very useful. For some larger SDPs one may also consider a Newton-augmented Lagrangian code with iterative linear algebra,

NCGAL [29]. LOQO is not quite competitive for conic optimization problems while for SOCP the other applicable codes as the corresponding benchmark in [20] shows perform quite comparably and, different from the situation for SDP, even for larger problem sizes.

Finally, no mention is made here of the meanwhile large pool of application-specific conic software, partly by the same authors, such as for sensor network localization, sparse minimization etc.

For additional information we refer to the related literature, for example, to a survey article with a similar scope [23] as well as to several articles in this volume.

## References

1. Andersen, M.S., Dahl, J., Vandenberghe, L.: Implementation of nonsymmetric interior-point methods for linear optimization over sparse matrix cones. *Mathematical Programming Computation* **2**(3–4), 167–201 (2010)
2. Andersen, E.D., Roos, C., Terlaky, T.: On implementing a primal-dual interior-point method for conic quadratic optimization. *Mathematical Programming (series B)* **95**, 249–277 (2003)
3. Armbruster, M., Fügenschuh, M., Helmberg, C., Martin, A.: A Comparative Study of Linear and Semidefinite Branch-and-Cut Methods for Solving the Minimum Graph Bisection Problem. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) *Integer Programming and Combinatorial Optimization*, Lecture Notes in Computer Science, pp. 112–124. Springer (2008)
4. Benson, H.Y., Vanderbei, R.J.: Solving Problems with Semidefinite and Related Constraints Using Interior-Point Methods for Nonlinear Programming. *Mathematical Programming Ser. B* **95**, 279–302 (2003)
5. Benson, S.J., Ye, Y.: Algorithm 875: DSDP5—software for semidefinite programming. *ACM Trans. Math. Software* **34**(3), Art. 16 (2008)
6. Borchers, B.: CSDP, A C library for semidefinite programming. *Optimization Methods and Software* **11**, 613–623 (1999)
7. Borchers, B., Young, J.: Implementation of a Primal-Dual Method for SDP on a Shared Memory Parallel Architecture. *Comp. Opt. Appl.* **37**, 355–369 (2007)
8. Burer, S., Monteiro, R.D.C.: A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming (series B)* **95**, 329–357 (2003)
9. Burer, S., Monteiro, R.D.C.: Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming (series A)* **103**, 427–444 (2005)
10. DIMACS 7th Challenge website, <http://dimacs.rutgers.edu/Challenges/Seventh/> (2000)
11. Fujisawa, K., Fukuda, M., Kobayashi, K., Kojima, M., Nakata, K., Nakata, M., Yamashita, M.: SDPA (SemiDefinite Programming Algorithm) User’s Manual — Version 7.0.5 Research Report B-448, Dept. of Math. and Comp. Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, January 2010.
12. Gijswijt, D.C., Mittelmann, H.D., Schrijver, A.: Semidefinite code bounds based on quadruple distances, *IEEE Transaction on Information Theorem*, to appear
13. Helmberg, C.: A Cutting Plane Algorithm for Large Scale Semidefinite Relaxations. In: Grötschel, M. (ed.) *The Sharpest Cut*, MPS-SIAM Series on Optimization, 233–256 (2004)
14. Helmberg, C., Rendl, F., Vanderbei, R.J., Wolkowicz, H.: An interior-point method for semidefinite programming, *SIAM Journal on Optimization* **6**, 342–361 (1996)
15. Helmberg, C., Rendl, F.: A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization* **10**, 673–696 (2000)
16. Helmberg, C., Kiwiel, K.C.: A Spectral Bundle Method with Bounds. *Mathematical Programming Ser. B* **93**, 173–194 (2002)

17. Kojima, M., Shindoh, S., Hara, S.: Interior-point methods for the monotone semidefinite linear complementarity problems. *SIAM Journal on Optimization* **7**, 86–125 (1997)
18. Mittelmann, H.D.: An independent benchmarking of SDP and SOCP solvers. *Mathematical Programming (series B)* **95**, 407–430 (2003)
19. Mittelmann, H.D.: Decision Tree for Optimization Software. <http://plato.la.asu.edu/guide.html> (2011)
20. Mittelmann, H.D.: Benchmarks for Optimization Software. <http://plato.la.asu.edu/bench.html> (2011)
21. Mittelmann, H. D., Vallentin, F.: High Accuracy Semidefinite Programming Bounds for Kissing Numbers. to appear in *Experimental Mathematics* (2010)
22. NEOS Server for Optimization. <http://www-neos.mcs.anl.gov/neos/> (2011)
23. Polik, I.: Conic optimization software. In *Encyclopedia of Operations Research*, Wiley (2010)
24. Sturm, J.F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software* **11**, 625–653 (1999)
25. Sturm, J.F.: Central region method, In: Frenk, J.B.G. Roos, C., Terlaky, T., Zhang, S. (eds.) *High Performance Optimization*, pp. 157–194. Kluwer Academic Publishers (2000)
26. Tütüncü, R.H., Toh, K.C., Todd, M.J.: Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming Ser. B* **95**, 189–217 (2003)
27. Yamashita, M., Fujisawa, K., Nakata, K., Nakata, M., Fukuda, M., Kobayashi, K., Goto, K.: A high-performance software package for semidefinite programs: SDPA 7. Research Report B-460, Dept. of Math. and Comp. Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152-8552, January 2010.
28. Ye, Y., Todd, M.J., Mizuno, S.: An  $O(\sqrt{n}L)$ -iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research* **19**, 53–67 (1994)
29. Zhao, X.-Y., Sun, D., Toh, K.-C.: A Newton-CG Augmented Lagrangian Method for Semidefinite Programming. *SIAM Journal on Optimization* **20**, 1737–1765 (2010)

# Chapter 24

## Latest Developments in the SDPA Family for Solving Large-Scale SDPs

**Makoto Yamashita, Katsuki Fujisawa, Mituhiro Fukuda,  
Kazuhiro Kobayashi, Kazuhide Nakata, and Maho Nakata**

### 24.1 Introduction

One of the key aspects in mathematical optimization is providing computer software packages to solve useful problems arising from practical situations. Software packages have always been an essential bridge between the theory of mathematical optimization and its application to other fields. The research on SemiDefinite

---

M. Yamashita (✉)

Department of Mathematical and Computing Sciences, Tokyo Institute of Technology,  
2-12-1-W8-29 Ookayama, Meguro-ku, Tokyo 152-8552, Japan  
e-mail: [Makoto.Yamashita@is.titech.ac.jp](mailto:Makoto.Yamashita@is.titech.ac.jp)

K. Fujisawa

Department of Industrial and Systems Engineering, Chuo University, 1-13-27 Kasuga,  
Bunkyo-ku, Tokyo, 112-8551, Japan  
e-mail: [fujisawa@indsys.chuo-u.ac.jp](mailto:fujisawa@indsys.chuo-u.ac.jp)

M. Fukuda

Department of Mathematical and Computing Sciences, Tokyo Institute of Technology,  
2-12-1-W8-41 Ookayama, Meguro-ku, Tokyo 152-8552, Japan  
e-mail: [mituhiro@is.titech.ac.jp](mailto:mituhiro@is.titech.ac.jp)

K. Kobayashi

Center for Logistics Research, National Maritime Research Institute, 6-38-1 Shinkawa,  
Mitaka-shi, Tokyo 181-0004, Japan  
e-mail: [kobayashi@nmri.go.jp](mailto:kobayashi@nmri.go.jp)

K. Nakata

Department of Industrial Engineering and Management, Tokyo Institute of Technology,  
2-12-1-W9-60 Ookayama, Meguro-ku, Tokyo 152-8552, Japan  
e-mail: [nakata.k.ac@m.titech.ac.jp](mailto:nakata.k.ac@m.titech.ac.jp)

M. Nakata

Advanced Center for Computing and Communication, RIKEN, 2-1 Hirosawa, Wako-city,  
Saitama, 351-0198 Japan  
e-mail: [maho@riken.jp](mailto:maho@riken.jp)

Programs (SDP) is no exception. Software packages for solving SDPs have often opened up new vistas for SDP researches and have extended the range of SDP applications.

Although many approaches were developed in the early stages of SDP package developments, they could solve only small SDPs and had unsatisfactory numerical accuracy. The situation completely changed in the mid 1990s, when it was shown that the primal-dual interior-point method (PDIPM) is effective in solving not only linear programming problems but also SDPs [1, 16, 19, 25, 30]. Nowadays, many software packages use PDIPM-based algorithms, such as SDPA [38], CSDP [7], SDPT3 [32], and SeDuMi [31]. In fact, such have become the main stream and are practical means of solving SDPs arising, for instance, from control theory [10] and graph theory [14].

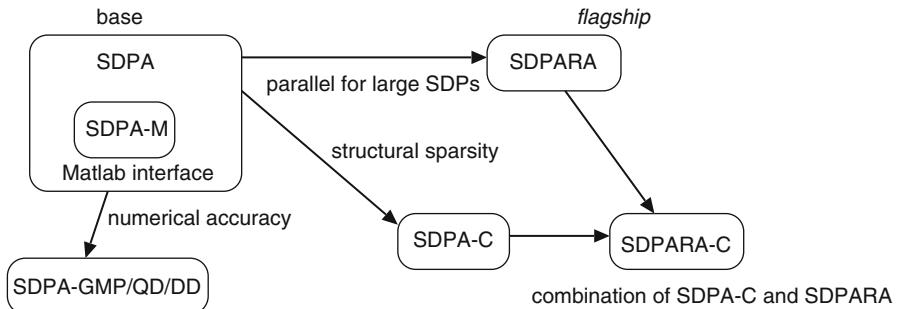
In the last decade, the range of SDP applications has expanded to ground-state energy computation in quantum chemistry [28, 29], Sensor Network Localization (SNL) problems [5], Polynomial Optimization Problems (POP) [20], etc. The SDPs of these applications tend to be extremely large-scale; some involve over 100,000 equality constraints.

On the other hand, the progress of computer technology through the last decade has been also remarkable. Cluster computer, parallel computing environments composed of multiple computing servers connected through high-speed network, are now providing enormous computing resources in many industrial and research fields. In addition, the most recent CPUs have multiple cores, which means parallel computing is becoming commonplace. In line with these trends, SDP software developers have sought to derive the best performance from these hardware by embedding theoretically backed results into software packages.

Many state-of-the-art features have been incorporated in latest SDPA developments. SDPA employs a Mehrotra type predictor–corrector PDIPM as its basic algorithm. As reported in [38], the latest version 7 uses multi-thread computing to avoid computational bottlenecks and shrink the total computation time, and it handles the sparse Schur complement matrix effectively. For instance, the authors of [38] compare the computation times of the latest version 7 and the second version released in 1996 in the same hardware environment. They report that the version 7 is more than 100 times faster than version 2 for most SDPs.

Over the last 15 years, we have developed variants of SDPA, which we have collectively named the *SDPA Family*. One of the reasons to expand the SDPA Family is to exploit the sparsity of SDPs. Furthermore, most SDPs have their own characteristics. For example, SDPs arising from graph theory have extremely sparse input data matrices and/or structural sparsity in the matrices. Another reason to expand the SDPA Family is to solve SDPs with high accuracy. SDPs in quantum chemistry sometimes should be solved with accuracies of  $10^{-8}$  or higher in order to be competitive with existing quantum chemical methods [28, 29].

We summarize the features of each software package and describe the ideas behind them in Sect. 24.3. Figure 24.1 illustrates how the packages in the SDPA Family are related to each other.



**Fig. 24.1** The SDPA family

- *SDPA* [38] is a software package for general SDPs that implements a Mehrotra type PDIPM. SDPA is the basis for the rest of the SDPA Family.
- *SDPA-M* is the Matlab interface for SDPA.
- *SDPA-GMP/DD/QD* are software packages which can find highly accurate solutions. SDPA-GMP/DD/QD utilize the GMP, DD and QD multiple precision libraries.
- *SDPA-C* implements the completion method in the PDIPM as a means to exploit structural sparsity in the input matrices [26].
- *SDPADA* [35] is the parallel version of SDPA. It is the flagship package of the SDPA Family. SDPADA is designed to solve extremely large-scale SDPs whose sizes are beyond the capabilities of other packages.
- *SDPADA-C* [27] is the parallel version of SDPA-C. It is a combination of SDPA-C and SDPADA.

The source codes and user's manuals of the SDPA Family are available at the SDPA project web page:

<http://sdpa.sourceforge.net>

Another noteworthy feature of the SDPA Family is that some packages are provided by the SDPA Online Solver [13]. Some SDPA users do not have their own cluster computer to execute SDPADA or SDPADA-C. The Online Solver enables such users to run the packages in the SDPA Family on a cluster computer via the Internet.

Powerful SDP software packages, including those of the SDPA Family, are becoming indispensable tools in many SDP applications. For example, an SDP approach [28, 29] for the ground-state energy computation in quantum chemistry relies on these state-of-art software packages.

This chapter is organized as follows. Section 24.2 describes the notations and the algorithmic framework of the PDIPM. Section 24.3 details each software package of the SDPA Family and the SDPA Online Solver. The subsections of this section range from the base package (SDPA) to the most complicated package (SDPADA-C). The subsections on SDPA-M, SDPA-GMP/QD/DD, and the SDPA Online Solver are

intended for newcomers to the SDPA Family. These subsections will be useful if you would like to use the SDPA Family as a black-box for your research. In particular, the SDPA Online Solver is a good starting point. We also include tips for Matlab users to use the SDPA Online Solver more conveniently. The subsections on SDPA, SDPARA, SDPA-C and SDPARA-C discuss the details of the algorithms and their implementation. These subsections should be interesting to readers who want to know how the SDPA Family solves SDPs faster. Section 24.4 shows numerical results of SDPARA running on a PC-cluster. We concentrate on SDPARA because it attains the highest performance for general SDPs. The numerical results confirm that SDPARA can solve extremely large-scale SDPs that are too large for other software packages to solve. In Sect. 24.5, we give closing remarks and discuss the future directions of the SDPA project.

## 24.2 Preliminaries

The section introduces the notations used in this chapter, and sets forth the basic framework of the PDIPM.

### 24.2.1 Primal-Dual Semidefinite Programs

The standard form of SDPs is of the following primal-dual form.

$$\text{SDP} \left\{ \begin{array}{l} \mathcal{P} : \text{minimize } \sum_{k=1}^m c_k x_k \\ \text{subject to } \mathbf{X} = \sum_{k=1}^m \mathbf{F}_k x_k - \mathbf{F}_0, \quad \mathbf{X} \succeq \mathbf{O}. \\ \mathcal{D} : \text{maximize } \mathbf{F}_0 \bullet \mathbf{Y} \\ \text{subject to } \mathbf{F}_k \bullet \mathbf{Y} = c_k \ (k = 1, \dots, m), \quad \mathbf{Y} \succeq \mathbf{O}. \end{array} \right. \quad (24.1)$$

We use  $\mathbb{S}^n$  for the space of  $n \times n$  symmetric matrices. The notation  $\mathbf{X} \succeq \mathbf{O}$  ( $\mathbf{X} > \mathbf{O}$ ) indicates that  $\mathbf{X} \in \mathbb{S}^n$  is a positive semidefinite (positive definite) matrix. The inner-product between  $\mathbf{U} \in \mathbb{S}^n$  and  $\mathbf{V} \in \mathbb{S}^n$  is defined by  $\mathbf{U} \bullet \mathbf{V} = \sum_{i=1}^n \sum_{j=1}^n U_{ij} V_{ij}$ .

It is common for most SDP applications that the input data matrices  $\mathbf{F}_0, \dots, \mathbf{F}_m$  share the same diagonal block structure  $(n_1, \dots, n_h)$ . To put it precisely, each input data matrix  $\mathbf{F}_k$  ( $k = 0, 1, \dots, m$ ) consists of sub-matrices in the diagonal positions as follows:

$$\mathbf{F}_k = \begin{pmatrix} \mathbf{F}_k^1 & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{F}_k^2 & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \ddots & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{F}_k^h \end{pmatrix}$$

where  $\mathbf{F}_k^1 \in \mathbb{S}^{n_1}, \mathbf{F}_k^2 \in \mathbb{S}^{n_2}, \dots, \mathbf{F}_k^h \in \mathbb{S}^{n_h}$ .

Note that  $\sum_{\ell=1}^h n_\ell = n$  and the variable matrices  $X$  and  $Y$  also share the same block structure. We define  $n_{\max}$  by  $\max\{n_1, \dots, n_h\}$ . For the blocks where  $n_\ell = 1$ , the constraints of positive semidefiniteness are equivalent to the constraints of the nonnegative orthant. Such blocks are sometimes called LP blocks.

The size of given SDPs can be roughly measured in terms of five numbers:

1.  $m$ : the number of equality constraints in the dual form  $\mathcal{D}$
2.  $n$ : the size of the matrices  $X, Y$  and  $F_k (k = 0, 1, \dots, m)$
3.  $n_{\max}$ : the size of the largest block of input data matrices
4.  $nnz$ : the total number of nonzero elements of all data matrices
5.  $sd$ : the ratio of nonzero elements of the Schur complement matrix

The definition of the last measure is given in Sect. 24.3.1. The latest version of SDPA solves SDPs in  $m \approx 100,000$  and  $n_{\max} \approx 2,000$ , and SDPARA solves them in  $m \approx 200,000$  and  $n_{\max} \approx 3,000$  when  $sd = 100\%$ . If  $sd < 10^{-3}\%$ , SDPARA takes  $m > 500,000$ .

### 24.2.2 Basic Framework of the Primal-Dual Interior-Point Method

Here, we present the basic framework of the PDIPM. One of the most important theoretical aspects of the PDIPM is that it solves both primal and dual forms simultaneously in polynomial-time.

The Karush–Kuhn–Tucker conditions guarantee that under Slater’s condition, the point  $(\mathbf{x}^*, \mathbf{X}^*, \mathbf{Y}^*)$  satisfying the following system is an optimal solution of (24.1).

$$\text{KKT} \begin{cases} \mathbf{X}^* = \sum_{k=1}^m \mathbf{F}_k \mathbf{x}_k^* - \mathbf{F}_0, & (\text{primal feasibility}) \\ \mathbf{F}_k \bullet \mathbf{Y}^* = c_k \quad (k = 1, 2, \dots, m), & (\text{dual feasibility}) \\ \sum_{k=1}^m c_k \mathbf{x}_k^* = \mathbf{F}_0 \bullet \mathbf{Y}^*, & (\text{primal-dual gap condition}) \\ \mathbf{X}^* \geq \mathbf{O}, \mathbf{Y}^* \geq \mathbf{O}. & (\text{positive semidefinite conditions}) \end{cases} \quad (24.2)$$

In each iteration, the PDIPM computes the search direction  $(d\mathbf{x}, d\mathbf{X}, d\mathbf{Y})$  from the current point towards the optimal solution and moves the current point along this search direction. If the point falls into a small neighborhood of the optimal solution, the PDIPM terminates and returns the approximate optimal solution. The PDIPM is described in many papers (See [1, 16, 19, 25, 30]). The framework we shall use relies on the HRVW/KSH/M approach [16, 19, 25], and we will use the appropriate norms  $\|\cdot\|$  for matrices and vectors for it.

### Algorithmic Framework of the PDIPM

Step 0: Choose an initial point  $(\mathbf{x}^0, \mathbf{X}^0, \mathbf{Y}^0)$  such that  $\mathbf{X}^0 > \mathbf{O}$  and  $\mathbf{Y}^0 > \mathbf{O}$ . Set the centering parameter  $\beta \in (0, 1)$ , the boundary parameter  $\gamma \in (0, 1)$ , the threshold parameter  $\epsilon > 0$ , and the iteration number  $s = 0$ .

Step 1: Evaluate the residuals of the primal feasibility  $\mathbf{P}$ , the dual feasibility  $\mathbf{d}$ , and the primal-dual gap  $g$  by

$$\begin{cases} \mathbf{P} = \mathbf{F}_0 - \sum_{k=1}^m \mathbf{F}_k x_k^s + \mathbf{X}^s, \\ d_k = c_k - \mathbf{F}_k \bullet \mathbf{Y}^s \quad (k = 1, \dots, m), \\ g = \sum_{k=1}^m c_k x_k^s - \mathbf{F}_0 \bullet \mathbf{Y}^s. \end{cases}$$

If  $\max\{\|\mathbf{P}\|, \|\mathbf{d}\|, |g|\} < \epsilon$ , output  $(\mathbf{x}^s, \mathbf{X}^s, \mathbf{Y}^s)$  as an approximate solution and stop the relation.

Step 2: Evaluate the Schur complement matrix  $\mathbf{B}$  by the formula

$$B_{ij} = ((\mathbf{X}^s)^{-1} \mathbf{F}_i \mathbf{Y}^s) \bullet \mathbf{F}_j. \quad (24.3)$$

Step 3: Apply the Cholesky factorization to  $\mathbf{B}$  and obtain the lower triangular matrix  $\mathbf{L}$  such that  $\mathbf{B} = \mathbf{L}\mathbf{L}^T$ .

Step 4: Obtain a component of the search direction  $d\mathbf{x}$ , by solving the following equations

$$\mathbf{L}\widetilde{\mathbf{d}\mathbf{x}} = \mathbf{r} \text{ and } \mathbf{L}^T d\mathbf{x} = \widetilde{d\mathbf{x}}$$

for the right-hand-side vector  $\mathbf{r}$  computed by

$$r_k = -d_k + \mathbf{F}_k \bullet ((\mathbf{X}^s)^{-1} (\mathbf{R} + \mathbf{P}\mathbf{Y}^s)) \quad (k = 1, \dots, m), \quad (24.4)$$

where  $\mathbf{R} = \beta\mu\mathbf{I} - \mathbf{X}^s \mathbf{Y}^s$  with  $\mu = \mathbf{X}^s \bullet \mathbf{Y}^s/n$ .

Step 5: Compute the remaining components of the search direction  $(d\mathbf{X}, d\mathbf{Y})$  as follows:

$$\begin{cases} d\mathbf{X} = -\mathbf{P} + \sum_{k=1}^m \mathbf{F}_k d\mathbf{x}_k \\ d\widetilde{\mathbf{Y}} = (\mathbf{X}^s)^{-1} (\mathbf{R} - d\mathbf{X}\mathbf{Y}^s), \quad d\mathbf{Y} = (d\widetilde{\mathbf{Y}} + d\widetilde{\mathbf{Y}}^T)/2. \end{cases}$$

Step 6: Evaluate the maximum step lengths that maintain positive definiteness.

$$\begin{cases} \alpha_P = \max\{\alpha \in [0, 1] : \mathbf{X}^s + \alpha d\mathbf{X} \succeq \mathbf{O}\} \\ \alpha_D = \max\{\alpha \in [0, 1] : \mathbf{Y}^s + \alpha d\mathbf{Y} \succeq \mathbf{O}\}. \end{cases}$$

Step 7: Update the current point by

$$(\mathbf{x}^{s+1}, \mathbf{X}^{s+1}, \mathbf{Y}^{s+1}) \leftarrow (\mathbf{x}^s, \mathbf{X}^s, \mathbf{Y}^s) + \gamma(\alpha_P d\mathbf{x}, \alpha_P d\mathbf{X}, \alpha_D d\mathbf{Y})$$

Set  $s \leftarrow s + 1$ , and return to Step 1.

Steps 2 and 3 take up most of the computation time, and both steps are related to the Schur Complement Matrix (SCM). Following the style of [37], we shall call these two steps ELEMENTS and CHOLESKY, respectively. As indicated in [37], these two bottleneck components often take up 80–90% of the entire computation time. Therefore, researchers have focused on reducing the times of these steps.

Another point is that theoretical convergence of the PDIPM does not necessarily ensure the convergence of the PDIPM in the software packages. It is difficult in practice for software packages to generate a sequence of points that accurately converges to the optimal solution. This problem is mainly because the computer architecture cannot avoid numerical errors caused by the limited storages of floating-point numbers. For example, the standard C/C++ *double* precision (IEEE 754) only handles approximately 16 digits. Software packages, including the SDPA Family, should add heuristic methods to the PDIPM to obtain the convergence of generated points for attaining sufficient numerical accuracy. Such heuristic methods often improve the numerical accuracy of double precision. However, some SDP applications require accuracies far beyond the allowable range of double precision. This difficulty motivated us to develop SDPA-GMP/QD/DD (Sect. 24.3.3), which use extended precision libraries to execute the PDIPM.

## 24.3 The SDPA Family

Here, we introduce the software packages of the SDPA Family and describe their theoretical foundations and implementations.

### 24.3.1 SDPA

SDPA is the basic software package of the SDPA Family for solving general SDPs. Its main algorithm is the Mehrotra type predictor–corrector PDIPM with the HRVW/KSH/M search direction [16, 19, 25] (see Sect. 24.2.2).

SDPA was the first software package to incorporate a technique for exploiting the sparsity of the ELEMENTS component [12]. In general, the elements of the SCM are evaluated by using  $B_{ij} = ((X^s)^{-1} \mathbf{F}_i Y^s) \bullet \mathbf{F}_j$  in Step 2 of the PDIPM. This formula requires two multiplications of two  $n \times n$  matrices and one computation of the inner-product of two  $n \times n$  matrices. However, if we focus only on non-zero elements of  $\mathbf{F}_i$  and  $\mathbf{F}_j$ , we can get another expression for the same formula, i.e.,

$$B_{ij} = \sum_{\gamma=1}^n \sum_{\epsilon=1}^n \left( \sum_{\alpha=1}^n \sum_{\beta=1}^n [X^{-1}]_{\gamma\alpha} [\mathbf{F}_i]_{\alpha\beta} Y_{\beta\epsilon} \right) [\mathbf{F}_j]_{\gamma\epsilon},$$

where  $[F_i]_{\alpha\beta}$  is the  $(\alpha, \beta)$  element of  $F_i$ . Let  $nz(F_i)$  be the set of indices of non-zero elements of  $F_i$ ,

$$nz(F_i) = \{(\alpha, \beta) : [F_i]_{\alpha\beta} \neq 0\},$$

and  $\#nz(F_i)$  be its cardinality. As shown in [12], the cost of  $B_{ij}$  based on the latter formula is almost proportional to  $(2 \times \#nz(F_i) + 1) \times \#nz(F_j)$ . If  $\#nz(F_i)$  and  $\#nz(F_j)$  are considerably less than  $n^2$  (namely,  $F_i$  and  $F_j$  are sparse), the latter formula is much cheaper than the original formula. For example, SDPs in graph theory often have very sparse input data matrices. In fact, most have only one or two nonzero elements. Exploiting sparsity has a prominent effect on such SDPs.

Depending on the sparsity of the input data matrices, SDPA automatically selects the best formula from among

$$\mathcal{F}_1 : B_{ij} = (X^{-1} F_i Y) \bullet F_j$$

for the case both  $F_i$  and  $F_j$  are dense.

$$\mathcal{F}_2 : B_{ij} = \sum_{(\gamma, \epsilon) \in nz(F_j)} \left( \sum_{\alpha=1}^n [X^{-1}]_{\gamma\alpha} [F_i Y]_{\alpha\epsilon} \right) [F_j]_{\gamma\epsilon},$$

for dense  $F_i$  and sparse  $F_j$ .

$$\mathcal{F}_3 : B_{ij} = \sum_{(\gamma, \epsilon) \in nz(F_j)} \left( \sum_{(\alpha, \beta) \in nz(F_i)} [X^{-1}]_{\gamma\alpha} [F_i]_{\alpha\beta} Y_{\beta\epsilon} \right) [F_j]_{\gamma\epsilon},$$

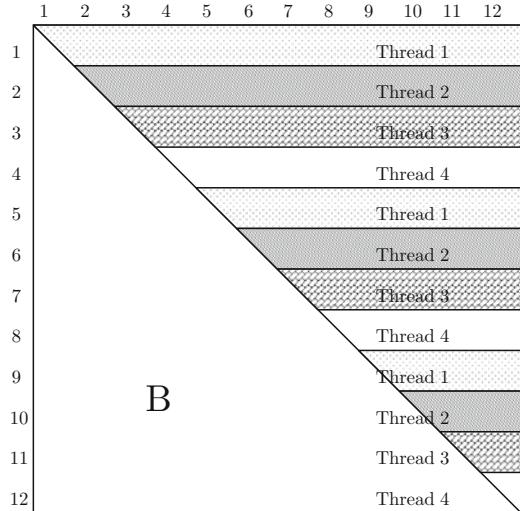
for the case both  $F_i$  and  $F_j$  are sparse.

Other software packages such as CSDP and SDPT3 use similar techniques for exploiting the sparsity of the ELEMENTS component.

A new feature of SDPA 7 is that it shrinks the computation time of ELEMENTS component even more by using multiple-threads [38]. Modern processors like Intel Xeon and AMD Opteron usually have multiple CPU cores. Assigning each thread to a core makes all the cores work in parallel. SDPA inherits the parallel concept of SDPARA, i.e., *row-wise distribution*. This concept was first used in SDPARA [37] for the parallel computation of the ELEMENTS component. Row-wise distribution is realized with the help of the MPI framework in SDPARA, whereas in SDPA, it is realized with multi-threading. The most important property of  $\mathcal{F}_1, \mathcal{F}_2$  or  $\mathcal{F}_3$  from the viewpoint of parallel computation is that the computations on each row are completely independent from those of the other rows.

Figure 24.2 shows a simple example of row-wise distribution, in which the SCM  $B$  is  $12 \times 12$  and there are four threads. Note that  $B$  is always symmetric in the PDIPM, so we only have to evaluate the upper triangular part. The second line is evaluated by the second thread and the seventh line by the third thread. A point here is that each thread can compute any row of  $B$  without having to communicate

**Fig. 24.2** The row-wise distribution



with the other threads. In practice, when  $u$  threads are available, we assign the first  $u$  rows to each thread at the beginning of the parallel evaluation. Then, we assign the remaining rows in sequential order to the threads which terminated their assigned rows, and continue this assignment until all the rows are computed. We call this assignment the *first-come-first-assigned row-wise distribution*. (For simplicity, Fig. 24.2 does not illustrate the first-come-first-assigned row-wise distribution but rather a simple cyclic assignment.)

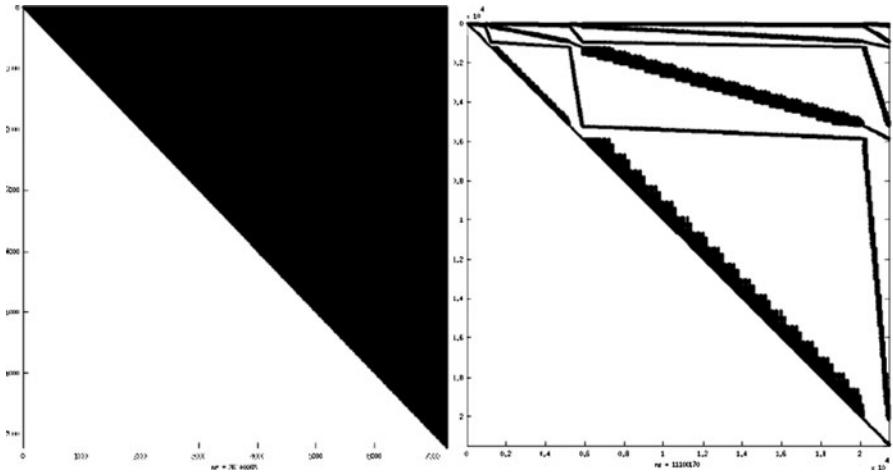
By using the row-wise distribution and  $u$  threads, SDPA can theoretically attain a speed-up of almost  $u$  times for ELEMENTS (see the numerical experiments on SDPARA in Sect. 24.4).

Another feature of SDPA 7 [38] is that it can handle sparse SCMs. In most SDPs of quantum chemistry and control theory, the SCM is fully dense. However, the SCM is sparse in some recent applications such as POP or SNL.

The non-zero pattern of  $\mathbf{B}$  is called *correlative sparsity* [18] and essentially defined by

$$\mathcal{S}_{\mathbf{B}} = \bigcup_{\ell=1}^h \{(i, j) \in \{1, 2, \dots, m\} : \mathbf{F}_i^\ell \neq \mathbf{O} \text{ and } \mathbf{F}_j^\ell \neq \mathbf{O}\}.$$

Throughout all iterations of the PDIPM,  $B_{ij} = 0$  if  $(i, j) \notin \mathcal{S}_{\mathbf{B}}$ . Hence, the non-zero pattern  $\mathbf{B}$  is fixed. Figure 24.3 shows a fully-dense SCM from quantum chemistry and a sparse SCM from POP. The common characteristic of POP and SNL is that they have a large number of diagonal block matrices  $(n_1, \dots, n_h)$  in the input data matrices. The number  $h$  of blocks sometimes exceeds 1,000 and  $\mathbf{F}_i^\ell$ , the  $\ell$ th block of  $\mathbf{F}_i$ , is empty for most  $\ell = 1, \dots, h$ . Therefore, the SCM corresponding to this  $\mathcal{S}_{\mathbf{B}}$  tends to be very sparse.



**Fig. 24.3** A fully-dense SCM from quantum chemistry, and a very sparse SCM from SNL

SDPA detects the pattern  $\mathcal{S}_B$  before starting the iterations, and decides whether the SCM is a sparse matrix or a dense matrix. We define the density of the SCM by

$$sd = \frac{\#\mathcal{S}_B}{m^2},$$

where  $\#\mathcal{S}_B$  is the cardinality of  $\mathcal{S}_B$ . When  $sd < 0.7$  holds, SDPA treats the SCM as a sparse matrix. If SDPA treats the SCM as sparse, it allocates a sparse type of memory storage to  $B$  and skips all computations of  $B_{ij}$  with  $(i, j) \notin \mathcal{S}_B$ . Then it uses sparse Cholesky factorization. The sparse memory storage and sparse Cholesky factorization greatly reduce the computation time needed to solve SDPs having a sparse SCM.

Until now, we have focused on the sparsity of the input data matrices and the SCM. The above implementations considerably reduce the total computation time. However, there still remains substantial computations involving dense matrices. For example, the above formula  $\mathcal{F}_1$  includes two dense matrix multiplications. SDPA (since version 6 [36]) and the SDPA Family utilize the BLAS/LAPACK library for dense matrix manipulations. These libraries take account of the characteristics of recent processors (e.g., CPU cache and memory bandwidth) and derive almost nearly peak performance from them. The SDPA Family calls the BLAS/LAPACK library for dense matrices and uses its own manipulations for sparse matrices. The ability to make an appropriate selection of either dense computation or sparse computation is a key advantage of the SDPA Family.

Another feature of SDPA is the callable library. SDPA is often executed as a stand-alone program. However, it can also be embedded in the users' own applications as a sub-module. The interface of the callable library is written in C++. The callable library is used in many applications, including data mining [21].

### 24.3.2 SDPA-M

SDPA-M is the Matlab interface of SDPA. Originally, SDPA-M was independent of SDPA. However, from version 6 onwards, SDPA-M has been modified to bridge the Matlab internal data structure and the SDPA callable library. Moreover, in version 7 [38], the source code of SDPA-M is merged into that of SDPA. Furthermore, version 7 supports SeDuMi-style input as well as SDPA-M style input.

SeDuMi-style input is given by the Matlab data  $A, b, c, K$  and parameters  $\text{par}$ . If users invoke SeDuMi by

```
[x,y,info] = sedumiwrap(A,b,c,K,pars);
```

they can also invoke SDPA-M by using the Matlab command

```
[x,y,info] = sedumiwrap(A,b,c,K,[],OPTION);
```

Note that the input data has the same format. The cost for the slight modification in the parameter set from SeDuMi-style  $\text{pars}$  to SDPA-style  $\text{OPTION}$  is trifling in comparison with the benefits brought by the more flexible selection in software packages. SeDuMi is apt to be more accurate than SDPA, whereas SDPA is several times faster than SeDuMi for most SDPs (see [38]). The SOCP cone  $K.q$  cannot be used for `sedumiwrap` now, since SDPA is only good for solving SDPs and LPs.

### 24.3.3 SDPA-GMP/DD/QD

Even though the PDIPM is a sophisticated algorithm for solving SDPs, it sometimes encounters numerical difficulties. A typical problem occurs when the SCM  $B$  becomes ill-conditioned near an optimal solution. In such case, the PDIPM may stop prematurely since it cannot obtain a correct Cholesky factorization of  $B$  in Step 3.

To overcome such numerical errors, SDPA employs multiple precision libraries GMP (GNU Multiple Precision Library),<sup>1</sup> DD (Double–Double) and QD (Quad–Double) [17]. The standard IEEE 754 double-precision has approximately 16 significant digits. The DD and QD libraries extend this accuracy to 32 digits and 64 digits, respectively. Furthermore, GMP can handle an arbitrary number of digits.

Of course, SDPA-GMP/QD/DD are much slower than SDPA. For instance, the computational cost of SDPA-GMP sometimes reaches 1,000 times that of SDPA. Incurred by this expensive computation cost, SDPA-GMP gives extremely accurate solutions. In SDPA-GMP, the threshold  $\epsilon$  at Step 1 in the PDIPM can be set to  $10^{-30}$  or even  $10^{-100}$ , whereas the default value of  $\epsilon$  is  $10^{-7}$  in SDPA.

Extremely high accuracy is needed to solve certain SDPs in POP [34] and quantum chemistry [28], and it is also needed for computing bounds for kissing

---

<sup>1</sup><http://gmplib.org/>.

number calculations [24]. Although computation times can be very long (more than a month), SDPA-GMP/QD/DD are indispensable software packages for these researches.

It is worth mentioning that the matrix manipulations of SDPA-GMP/QD/DD are those of MPACK.<sup>2</sup> MPACK is a multiple precision arithmetic version of BLAS and LAPACK. MPACK is a by-product of the SDPA Family, and it is available as a stand-alone package so that users can directly employ its functions.

#### 24.3.4 SDPA-C

The main purpose of SDPA-C is to exploit structural sparsity in the input data matrices.

SDPA-C employs the completion method of [26]. Here, we briefly introduce the basic idea behind this method. Let us consider the following simple dual SDP of the variable matrix  $\mathbf{Y} \in \mathbb{S}^6$ :

$$\max : \mathbf{F}_0 \bullet \mathbf{Y} \quad \text{s.t.} : \mathbf{F}_1 \bullet \mathbf{Y} = 5, \quad \mathbf{Y} \succeq \mathbf{O}.$$

where

$$\mathbf{F}_0 = \begin{pmatrix} -2 & -3 & & 1 \\ & 4 & -2 & -5 \\ -3 & -2 & 7 & -1 \\ & & -1 & 3 & -6 \\ -5 & & & -2 \\ 1 & & -6 & 5 \end{pmatrix} \quad \text{and} \quad \mathbf{F}_1 = \begin{pmatrix} 3 & 3 & -1 \\ 9 & & \\ 3 & 1 & -2 & 4 \\ & -2 & 2 & 2 \\ 4 & 7 & & \\ -1 & 2 & 3 \end{pmatrix}.$$

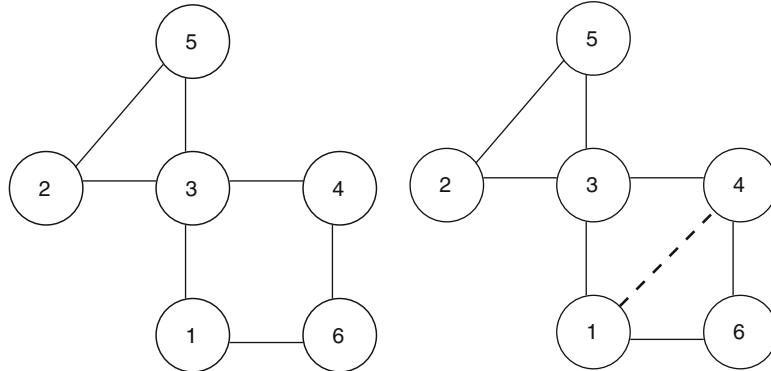
To detect the structural sparsity, we first define the *aggregated sparsity pattern*  $\mathcal{A} \in \mathbb{S}^n$  by  $\mathcal{A}(i, j) = *$  if  $[\mathbf{F}_k]_{ij} \neq 0$  for some  $k = 0, 1, \dots, m$ , where \* denotes the non-zero elements. In the above example, the aggregated sparsity pattern is

$$\mathcal{A} = \begin{pmatrix} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & * & & * & & * & \\ 2 & & * & * & & * & \\ 3 & * & * & * & * & * & \\ 4 & & * & * & & * & \\ 5 & * & * & & * & & \\ 6 & * & & * & & * & \end{pmatrix}.$$

From the definition of the inner-product, not all elements of  $\mathbf{Y}$  are needed to evaluate the objective function and equality constraints in the dual SDP  $\mathcal{D}$ . The minimal set of elements of  $\mathbf{Y}$  is  $\{Y_{ij} : (i, j) \in \mathcal{A}\}$ . However, all elements of  $\mathbf{Y}$  must be used to check whether  $\mathbf{Y} \succeq \mathbf{O}$ .

---

<sup>2</sup><http://sourceforge.net/projects/mplapack/>.



**Fig. 24.4** Aggregated sparsity pattern (*left*) and extended sparsity pattern (*right*)

The simple question we address here is how to find the minimal elements of  $\mathbf{Y}$  to keep  $\mathbf{Y} \geq \mathbf{O}$  covering  $\mathcal{A}$ . It is shown in [15] that the minimal set of elements in  $\mathbf{Y}$  can be obtained from the characteristics of the chordal graph. A graph is said to be chordal if every cycle of length  $\geq 4$  has a chord, an edge joining two nonconsecutive vertices of the cycle. The left figure of Fig. 24.4 shows the graph corresponding to  $\mathcal{A}$  in the above example, where each node  $i$  corresponds to the  $i$ th row/column of  $\mathcal{A}$  and an edge  $(i, j)$  is added if  $\mathcal{A}(i, j) = *$ .

The graph at the left in Fig. 24.4 is not chordal because the cycle  $1 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 1$  has no chord. To make the graph chordal, we add an edge  $(1, 4)$ . The resulting graph is shown at the right of Fig. 24.4. The *extended sparsity pattern*  $\mathcal{E}$  of the matrix  $\mathcal{A}$  is the pattern matrix

$$\mathcal{E} = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \left( \begin{array}{cccccc} * & & * & + & & * \\ * & * & & & * & \\ * & * & * & * & * & \\ + & & * & * & & * \\ * & * & & & * & \\ * & & * & & * & \end{array} \right) \end{matrix}$$

whose  $(i, j)$  element is nonzero if edge  $(i, j)$  exists in the chordal graph corresponding to  $\mathcal{A}$ . To reveal its hidden structure, we permute the rows and columns of  $\mathcal{E}$ .

$$\mathcal{E} = \begin{matrix} & \begin{matrix} 2 & 5 & 3 & 1 & 4 & 6 \end{matrix} \\ \begin{matrix} 2 \\ 5 \\ 3 \\ 1 \\ 4 \\ 6 \end{matrix} & \left( \begin{array}{cccccc} * & * & * & & & \\ * & * & * & & & \\ * & * & * & * & * & \\ * & * & * & + & * & \\ * & + & * & * & * & \\ * & * & * & * & * & \end{array} \right) \end{matrix}.$$

The above permutation makes it clear that the right figure of Fig. 24.4 can be decomposed into three cliques induced by nodes  $C_1 = \{2, 5, 3\}$ ,  $C_2 = \{3, 1, 4\}$  and  $C_3 = \{1, 4, 6\}$ . A subset of nodes in a graph is said to be a clique if every pair of nodes in the set is connected by an edge.

The completion method [26] indicates that we can decompose  $\mathbf{Y}$  into sub-matrices based on these cliques. In other words, we can execute the PDIPM while keeping the positive definiteness of three sub-matrices,  $\mathbf{Y}_1$ ,  $\mathbf{Y}_2$ , and  $\mathbf{Y}_3$ , instead of the whole matrix  $\mathbf{Y}$ :

$$\mathbf{Y}_1 = \begin{pmatrix} Y_{22} & Y_{25} & Y_{23} \\ Y_{25} & Y_{55} & Y_{35} \\ Y_{23} & Y_{35} & Y_{33} \end{pmatrix}, \mathbf{Y}_2 = \begin{pmatrix} Y_{33} & Y_{13} & Y_{34} \\ Y_{13} & Y_{11} & Y_{14} \\ Y_{34} & Y_{14} & Y_{44} \end{pmatrix}, \mathbf{Y}_3 = \begin{pmatrix} Y_{11} & Y_{14} & Y_{16} \\ Y_{14} & Y_{44} & Y_{46} \\ Y_{16} & Y_{46} & Y_{66} \end{pmatrix}.$$

From the definition of the aggregate sparsity pattern, it is clear that the sub-matrices cover all elements of  $\mathbf{Y}$  necessary for making the evaluation of the objective function and the equality constraints.

We can recover the entire matrix  $\mathbf{Y} \geq \mathbf{O}$  from  $\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_3 \geq \mathbf{O}$ . This step is called *the matrix completion*. After solving SDPs with smaller sub-matrices, we perform the matrix completion on  $\mathbf{Y}_1^*, \mathbf{Y}_2^*, \mathbf{Y}_3^*$  to obtain the final result  $\mathbf{Y}^*$ .

We described above the basic idea behind the completion method from the viewpoint of the dual SDP. Let us now assume that the sizes of the sub-matrices of  $\mathbf{Y}$  are  $(n_1, \dots, n_h)$ . Accordingly, the computation cost of the Cholesky factorization of  $\mathbf{Y}$  goes from  $n^3$  to  $\sum_{\ell=1}^h n_\ell^3$ . Certain computation costs related to  $\mathbf{Y}$  are evaluated in the same way. Furthermore, other parts of the algorithm, such as the ELEMENTS component, are affected (see Sect. 24.3.6). Hence, if the sizes  $(n_1, \dots, n_h)$  of the sub-matrices are somewhat smaller than  $n$ , the completion method can be used to solve SDPs very efficiently.

Note that the addition of two edges  $(1, 4)$  and  $(3, 6)$  in the above example leads to another pair of cliques  $\bar{C}_1 = \{2, 5, 3\}$  and  $\bar{C}_2 = \{3, 1, 4, 6\}$ . There are multiple ways to choose cliques and the choice of cliques affects the computation cost. Since finding the best cliques is known to be an NP-hard problem, SDPA-C utilizes the meta-heuristics of SPOOLES<sup>3</sup> and/or METIS.<sup>4</sup>

SDPA-C solves SDPs faster than SDPA does when the SDPs have an underlying structural sparsity pattern related to “nice” chordal graphs. We encounter such SDPs in SDP relaxations for combinatorial optimization problems.

### 24.3.5 SDPARA

SDPARA (SemiDefinite Programming Algorithm paRALlel version) is the flagship software package among the SDPA Family because of its efficiency in solving

---

<sup>3</sup><http://www.netlib.org/linalg/spooles/>.

<sup>4</sup><http://www.cs.umn.edu/~metis>.

extremely large-scale SDPs. SDPARA is designed to execute the PDIPM on a parallel computer with a distributed memory space. The speed-up realized by SDPARA essentially owes to its use of parallel computation to overcome the computational bottlenecks of ELEMENTS and CHOLESKY components.

In this subsection, we focus on these two components. The latest version of SDPARA [35] uses different parallel computation depending on the density of the SCM. We shall first take a look on the parallel computation for SDPs having fully-dense SCMs and then move on to the case of sparse SCMs. We assume that a parallel computer is composed of multiple nodes and each node has multiple processors and its own memory space. Since each node has its own independent memory space, network communication is necessary for the node to access the memory space of another node. In contrast, multiple CPU cores on the same nodes usually share the memory space; hence, there is no network communication between multiple threads.

Suppose that  $u$  nodes are available on the parallel computer. Then, within the simple framework of the row-wise distribution illustrated in Fig. 24.2, the  $p$ th node evaluates the rows of  $\mathbf{B}$  that have been assigned to it in the cyclic manner. More specifically, the  $p$ th node evaluates the rows in the set  $\mathcal{R}_p$  defined by

$$\mathcal{R}_p = \{i : (i - p)\%u = 0, \quad 1 \leq i \leq m\},$$

where  $a\%b$  is the remainder of the division of  $a$  by  $b$ .

The memory access strategy of SDPARA is different from that of SDPA. One difference is the duplication strategy of input matrices and variables. As pointed in Sect. 24.3.1, row-wise distribution requires no communication between nodes, because  $\mathcal{R}_p$  and  $\mathcal{R}_q$  can be evaluated independently when  $p \neq q$ . To exploit this node independence, we duplicate all input data matrices  $\mathbf{F}_0, \mathbf{F}_1, \dots, \mathbf{F}_m$  on all nodes before starting the iterations of the PDIPM, and  $X^{-1}$  and  $\mathbf{Y}$  are updated on all nodes at every iteration.

Another difference between SDPA and SDPARA is the different scheme of allocating rows of the SCM to the nodes. In SDPARA, the rows in  $\mathcal{R}_p$  are assigned to the  $p$ th node. This assignment is fixed during the iterations in the PDIPM, since if we apply the first-come-first-assigned row-wise distribution scheme as in SDPA to all the nodes, it becomes extremely difficult to predict the node in which the evaluated rows are stored and the redistribution for CHOLESKY component becomes expensive.

On the other hand, inside the  $p$ th node, a multi-thread computation is used to evaluate the rows in  $\mathcal{R}_p$  as in SDPA, because threads on the same node can share the memory space. In this sense, SDPARA implements the hybrid strategy of row-wise distribution and first-come-first-assigned distribution. We call this the hybrid row-wise distribution in the ELEMENTS component of the algorithm.

After the evaluation of the ELEMENTS component, SDPARA proceeds to the CHOLESKY component. For a dense SCM, SDPARA employs the parallel Cholesky factorization of ScaLAPACK [6]. In ScaLAPACK, the matrix to be factorized should be distributed with two-dimensional block-cyclic distribution (TDBCD) over multiple nodes. Figure 24.5 shows the matrix of Fig. 24.2 distributed

**Fig. 24.5** Two-dimensional Block-Cyclic distribution

	1	2	3	4	5	6	7	8	9	10	11	12
1	N1		N2		N1		N2		N1		N2	
2												
3	N3		N4		N3		N4		N3		N4	
4												
5	N1		N2		N1		N2		N1		N2	
6												
7	N3		N4		N3		N4		N3		N4	
8												
9	N1		N2		N1		N2		N1		N2	
10												
11	N3		N4		N3		N4		N3		N4	
12												

by TDBCD. The elements  $B_{15}$  and  $B_{26}$  are stored on the first node, while  $B_{57}$  and  $B_{58}$  are on the second node. Reference [11] describes how TDBCD improves the performance of the parallel Cholesky factorization. SDPARA distributes the matrix over multiple nodes by using row-wise distribution. Hence, SDPARA should redistribute the elements of the matrix before calling the parallel Cholesky factorization routine. The network communication cost overhead for this redistribution is small and should have little effect on efficiency.

Next, let us consider the case of a sparse SCM. Since a sparse SCM is much more complicated to handle, we shall describe only the fundamental concepts. An interested reader should consult [35] for a more through discussion.

Applying the row-wise distribution directly to a sparse SCM does not work so well, because the computation costs of the rows are somewhat unbalanced. In the formula-cost-based distribution for the sparse SCM  $\mathbf{B}$ , we first estimate the cost for each nonzero element  $B_{ij}$  on the basis of the selection of  $\mathcal{F}_1, \mathcal{F}_2$  or  $\mathcal{F}_3$ . We distribute the nonzero elements of  $\mathbf{B}$  to all nodes so that the computation loads on all participating nodes will be appropriately balanced.

The CHOLESKY component should follow the ELEMENTS component. SDPARA uses the MUMPS library [2–4] for sparse Cholesky factorization. The memory distribution of the ELEMENT component can be used because MUMPS can use it directly without redistribution. In other words, a sparse SCM does not require redistribution, and hence, we can avoid the network communications.

SDPARA utilizes parallel computations for both dense and sparse SCMs and can solve extremely large-scale SDPs that other software packages cannot solve. In addition, SDPARA has high scalability. Numerical results are reported in Sect. 24.4.

### 24.3.6 SDPARA-C

SDPARA-C [27] integrates the completion method implemented in SDPA-C and the parallel computation implemented in SDPARA. SDPARA-C is for solving extremely large-scale SDPs that have structurally sparse input data matrices.

SDPA-C uses a different way of evaluating elements in the SCM instead of  $\mathcal{F}_1, \mathcal{F}_2$  and  $\mathcal{F}_3$ , and so does SDPARA-C. The simple row-wise distribution of SDPARA is replaced with a *hashed row-wise distribution*.

The principal characteristics of the completion method in the framework of the PDIPM is that the variable matrices  $X^{-1}$  and  $Y$  are stored with their Cholesky factors  $M$  and  $N$ , respectively. More precisely, they are stored as the lower triangular matrices  $M$  and  $N$  such that  $X = MM^T$  and  $Y^{-1} = NN^T$ . This enables us to work with the chordal graph easily. The matrix  $N$  inherits the structural sparsity of the input data matrices and, hence, is sparser than  $Y$  itself.

We define by  $e_k \in \mathbb{R}^n$  the vector whose  $k$ th element is 1 and whose other elements are 0 and let  $[F_i]_{*k}$  be the  $k$ th column of  $F_i$ . Accordingly, the formula for evaluating the  $(i, j)$  element of the SCM can be reformulated as

$$\begin{aligned} B_{ij} &= (X^{-1}F_iY) \bullet F_j = (X^{-1}F_jY) \bullet F_i = \text{Tr}X^{-1}F_jYF_i \\ &= \sum_{k=1}^n e_k^T X^{-1}F_jYF_i e_k = \sum_{k=1}^n e_k^T (MM^T)^{-1}F_j(NN^T)^{-1}F_i e_k \\ &= \sum_{k=1}^n (M^{-T}M^{-1}e_k)^T F_j(N^{-T}N^{-1}[F_i]_{*k}). \end{aligned}$$

The second equality holds because of the symmetry of  $B$ .

As pointed out above,  $N$  becomes sparse because it inherits the sparsity of data matrices  $F_1, \dots, F_m$ . In contrast,  $N^{-1}$  becomes fully dense. To exploit the sparsity of  $N$ , we solve  $Nz = [F_i]_{*k}$  instead of multiplying  $[F_i]_{*k}$  by  $N^{-1}$ . During each iteration of the PDIPM, most of the computation time is spent in solving the triangular system. The principal issue in parallel computation is how the SCM is distributed among the nodes. As we described earlier, we use hashed row-wise distribution in SDPARA-C. The computational cost for the  $i$ th row is almost proportional to the number of non-zero columns of  $F_i$  because the computation for  $M^{-T}M^{-1}e_k$  for  $k = 1, \dots, n$  has to be done only once each iteration. We use the symbol  $\text{NZC}_i \subset \{1, \dots, n\}$  to denote the set of non-zero columns of  $F_i$ , and  $\#\text{NZC}_i$  for its cardinality. Note that SDPARA-C assumes that the SDP to be solved is structurally sparse. Because of this,  $\#\text{NZC}_i$  in  $i = 1, \dots, m$  is sometimes very non-uniform. Accordingly, parallel efficiency suffers when using row-wise distribution. For example, some SDPs in graph theory, e.g., graph partitioning problems, have  $\#\text{NZC}_1 = m$  and  $\#\text{NZC}_2 = \dots = \#\text{NZC}_m = 1$ . In this case, the cost for the first row is  $m$  times larger than that of any other row.

We divide the row indices  $\{i : 1 \leq i \leq m\}$  of  $\mathbf{B}$  to two subsets  $\mathcal{U}$  and  $\mathcal{V}$ :

$$\mathcal{U} = \{i : 1 \leq i \leq m, \#NZC_i > \text{thres}\} \quad \text{and} \quad \mathcal{V} = \{i : 1 \leq i \leq m\} \setminus \mathcal{U}$$

with a fixed threshold  $\text{thres}$  depending on  $m$  and  $n$ . The rows with indices in  $\mathcal{U}$  require a high computation cost for evaluating the corresponding elements of the SCM. In hashed row-wise distribution, all nodes are assigned to rows with indices in  $\mathcal{U}$ . The rows with indices in  $\mathcal{V}$  are evaluated in the same way as in row-wise distribution. Let us suppose that  $\text{NZC}_i = \{k_1^i, k_2^i, \dots, k_{\#NZC_i}^i\}$  for  $i \in \mathcal{U}$  and the number of available nodes are  $u$ . Then the  $p$ th node evaluates part of  $B_{ij}$

$$B_{ij}^p = \sum_{k_q^i : (q-p) \% u = 0} (\mathbf{M}^{-T} \mathbf{M}^{-1} e_{k_q^i})^T \mathbf{F}_j (\mathbf{N}^{-T} \mathbf{N}^{-1} [\mathbf{F}_i]_{*k_q^i})$$

such that  $B_{ij} = \sum_{p=1}^u B_{ij}^p$ . After all nodes terminate their own part of the  $i$ th row, the results are sent to the  $p$ 'th node such that  $i \in \mathcal{R}'_p$  and the  $(i, j)$  element are obtained from  $B_{ij} = \sum_{p=1}^u B_{ij}^p$ . This procedure is repeated for all  $i \in \mathcal{U}$ .

The CHOLESKY component is basically the same as it is in SDPARA. In other words, we apply parallel Cholesky factorization of ScaLAPACK to the SCM  $\mathbf{B}$ .

The numerical results of [27] shows that SDPARA-C is efficient for SDPs that have many equality constraints and structurally sparse input data matrices.

### 24.3.7 SDPA Online Solver

The SDPA Online Solver [13] itself is not a software package for SDPs. Nonetheless, we believe that SDP researchers would benefit from this introduction to the SDPA Online Solver.

Even though the SDPA Family is composed of free software packages and distributed under GPL (GNU Public License), it still seems to have two main obstacles to using them to solve extremely large-scale SDPs. The first obstacle is the complicated installation procedure. Although SDPA has becomes much simpler to install as a result of adding a configure script to version 7.0, the compilation or installation of optimized BLAS libraries (e.g., ATLAS, Intel MKL, AMD ACML and so on) remains as a difficult step for users who are not familiar with optimized libraries. In particular, computer hardware expertise is required to derive peak performance from optimized libraries. In addition, users need to update software packages for every release of the packages. The second obstacle is that users need to set up parallel computers to use SDPARA or SDPARA-C. Even though the computer hardware has become much cheaper, most users do not want to buy and maintain their own PC-clusters.

The SDPA Online Solver is a powerful tool that removes these obstacles. It provides its own parallel computer, so the users will be freed from bother of setting up and maintaining parallel computers. Furthermore, it regularly updates the SDPA Family.

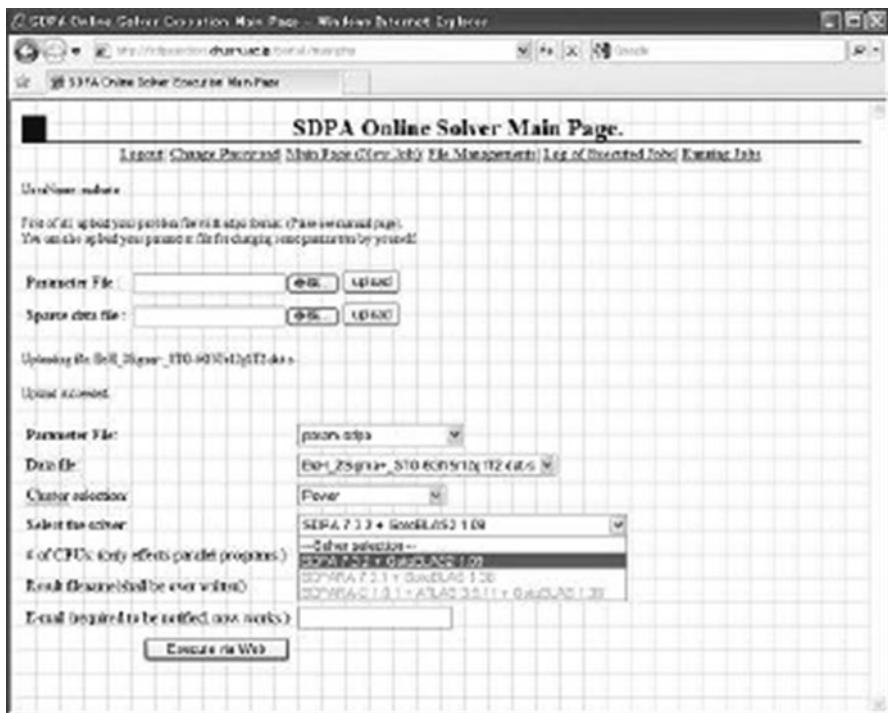


Fig. 24.6 Interface of SDPA online solver

The steps involved using the SDPA Online Solver are very simple. (1) Users register their information to create accounts at the Online Solver website. (2) They send their own SDPs in the *SDPA sparse format* via their web browser. (3) They select software packages (SDPA, SDPARA, or SDPARA-C) and parameters and push the “Execute via Web” button. (4) The Solver automatically invokes the selected package on the parallel computer. (5) Users receive the result through their web browser and/or e-mail. Figure 24.6 shows the interface of the SDPA Online Solver (steps (2) and (3)).

As described in (3), the Solver accepts the SDPA sparse format, a standard format for SDP problems. For example, the standard benchmark problems [8, 23] collects SDPs in SDPA sparse format.

Here, we give Matlab users a few tips for SDPA Online Solver. Since the Solver supports only the SDPA sparse format, some users should transform their SDPs into this format. If you use YALMIP [22], which is an excellent Matlab toolbox to convert certain optimization problems into the standard format, the function `savesdpafile` will convert the YALMIP SDP model into the SDPA sparse format. Another data format is the SeDuMi input format. As noting in Sect. 24.3.2, the

SeDuMi input format is expressed by the quadruple,  $A, b, c, K$ . The `SedumiToSDPA` function of SDPA-M converts the data into the SDPA sparse format.

```
SedumiToSDPA('input.dat-s',A,b,c,K,'%8.16e');
```

The first argument '`input.dat-s`' is the name of the file into which the data will be converted, and the last argument sets the output format of floating-point numbers in the style of the `printf` function of the C/C++ language. Although the SDPA Online Solver cannot accept YALMIP or SeDuMi input format directly because of Matlab limitations, these tips will be useful to users whose SDPs are not in the SDPA sparse format but in the YALMIP or SeDuMi format.

## 24.4 Numerical Results of SDPARA

We shall focus on SDPARA for three reasons. The first reason is that SDPARA is a general SDP software package designed to solve extremely large-scale SDPs with the help of parallel computation. SDPARA can solve the largest general SDPs in the SDPA Family in the shortest time. Next, many ideas and implementation techniques are incorporated in SDPARA. Finally, SDPARA is now available via the SDPA Online Solver. Therefore, showing the performance of SDPARA will be beneficial for the users of the SDPA Online Solver to estimate the computation times of their own SDPs. Exhaustive experiments on SDPARA have been conducted in the latest report [35].

Reference [38] describes extensive numerical experiments on the latest versions of SDPA and SDPA-GMP, and [27] gives results on SDPARA-C. SDPA-M gives essentially the same result as SDPA, except for subtle numerical errors, since SDPA-M internally calls the callable library of SDPA.

We executed all numerical experiments on a PC-cluster composed of 16 nodes connected by a Myrinet-10G network. Each node had two processors (Xeon X5460, 3.16 GHz) and a memory space of 48 GB, and each processor had four CPU cores.

The SDPs to which we applied SDPARA are summarized in Table 24.1. We selected the SDPs from three types of applications. The first type “QC” comprised SDPs in quantum chemistry to estimate the electrical structure of some

**Table 24.1** SDP sizes

Type	Name	$m$	$n$	$n_{\max}$	nnz	sd (%)
QC	B.2P.DZ.pqgt1t2	7,230	5,990	1,450	1,683,900	100
QC	CH2.3B1.DZ.pqgt1t2	27,888	15,914	4,032	9,438,380	100
SDPLIB	Control11	1,596	165	110	829,070	100
SDPLIB	ThetaG51	6,910	1,001	1,001	54,182	100
POP	OptControl(20,8,5,0)	109,246	33,847	136	1,497,271	4.39
POP	OptControl(20,8,6,0)	149,903	46,272	153	2,322,380	4.21
POP	Nonscomp(30,000)	299,990	509,986	6	1,979,939	7.05e-03

atoms/molecules [28]. The second type “SDPLIB” comprised SDPs from the standard SDP benchmark library, SDPLIB [8]. The third type “POP” comes from SDP relaxations of POPs generated by SparsePOP [33]. SparsePOP generates the SDP relaxation of a POP and outputs the generated SDP as an SDPA sparse format file with the appropriate option. For example, executing `sparsePOP('optControl(20,8,5,0)')` generates the SDP relaxation of `optControl(20,8,5,0)`. The SDPs `nonscomp(30,000)` is the largest SDP which we can generate from sparsePOP on a computer with 48 GB of memory.

The representative sizes of each SDP in Table 24.1 are  $m, n, n_{\max}, nnz$  and  $sd$  (see Sect. 24.2.1). Note that SDPs of the “QC” and “SDPLIB” types usually have fully-dense SCMs, as indicated by  $sd = 100\%$ , while SDPs of “POP” type usually have sparse SCMs.

We shall discuss the numerical results for B.2P.DZ.pqgt1t2 and `optControl(20,8,5,0)`, starting from the performance on a single node to show the improvements of SDPA.

The formulas  $\mathcal{F}_1, \mathcal{F}_2$  and  $\mathcal{F}_3$ , as implemented in SDPA and SDPARA, are keys to solving QC-SDPs in a practical amount of time. By appropriately choosing  $\mathcal{F}_1, \mathcal{F}_2$  or  $\mathcal{F}_3$  to evaluate each column of the SCM, we can process the ELEMENTS component efficiently. The formula  $\mathcal{F}_3$  are mainly used for B.2P.DZ.pqgt1t2, since most of the input matrices  $F_1, \dots, F_m$  are very sparse. More specifically, the average density of  $F_1, \dots, F_m$  is only 0.0045%. However,  $\mathcal{F}_1$  is more effective than  $\mathcal{F}_3$  for certain matrices that are considerably denser.

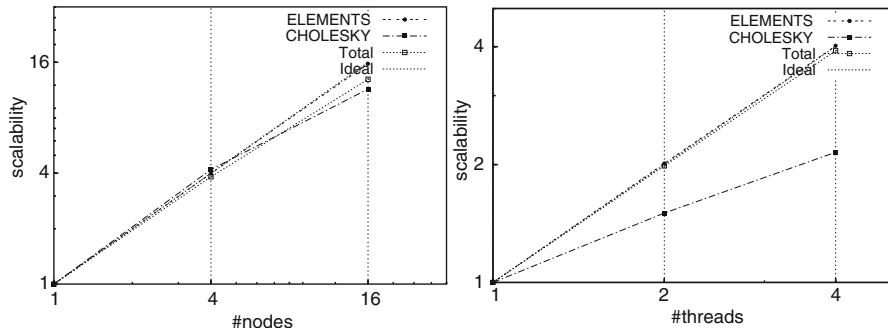
On the other hand, the key to solving `optControl(20,8,5,0)` is exploiting the sparsity of the SCM. The SDP has 109,246 constraints ( $m = 109,246$ ). Since the size of the SCM is  $m \times m$ , the memory space required to store the SCM in the fully-dense format would be 89 GB. Furthermore, `nonscomp(30,000)` have  $m = 299,990$  constraints, so it would occupy 65 TB of memory to store it in the fully-dense format. Such memory space requirements are prohibitive. The SCMs of these problems are sparse. In the case of `optControl(20,8,5,0)`,  $sd = 4.39\%$ , the required memory space to store the SCM in the sparse format is approximately 4 GB, and recent PCs can meet this requirement. In addition, skipping the computations for the zero elements in the SCM shortens the computation time of the ELEMENTS component. Moreover, applying sparse Cholesky factorization to the sparse SCM appreciably reduces the time taken by the CHOLESKY component.

Next, let us look at the effect of the parallel computation of SDPARA. Table 24.2 shows the computation times (in seconds) of SDPARA to solve the SDPs while varying the number of nodes and the number of threads. One of the measures to evaluate a parallel software is scalability, the ratio between the time taken to solve a problem by sequential computation and the time to solve it by parallel computation. For example, `optControl(20,8,5,0)` is solved in 5284.65 s on 1 node/1 thread and 1074.20 s on 16 nodes/1 thread; hence, its scalability is 4.91.

Figure 24.7 shows the scalability of B.2P.DZ.pqgt1t2 with respect to the number of nodes (left figure) and threads (right figure). The left figure indicates the high scalability (12.94) in going from 1 to 16 nodes with 1 thread. In particular, the ELEMENT component scores 15.70, which is very close to the ideal

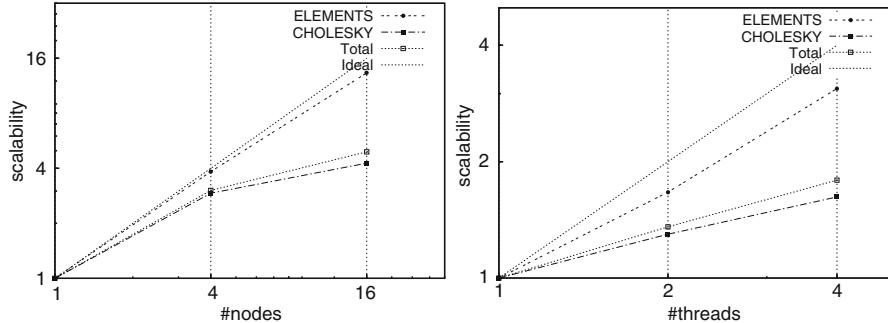
**Table 24.2** Computation time of SDPARA on selected SDPs in seconds

Name	#Threads	#Nodes	1	4	16
B.2P.DZ.pqgt1t2	1	ELEMENTS	28678.56	7192.87	1826.37
		CHOLESKY	548.99	131.09	47.90
		Total	29700.98	7764.18	2294.76
	2	ELEMENTS	14252.88	3600.96	913.31
		CHOLESKY	365.19	89.92	35.02
		Total	14918.20	3956.62	1203.50
	4	ELEMENTS	7147.63	1798.53	460.43
		CHOLESKY	255.47	55.30	27.26
		Total	7613.13	2034.93	647.55
optControl(20,8,5,0)	1	ELEMENTS	1137.58	296.31	85.53
		CHOLESKY	4053.63	1386.39	950.09
		Total	5284.65	1744.47	1074.20
	2	ELEMENTS	682.03	165.22	71.19
		CHOLESKY	3122.31	1020.88	818.99
		Total	3895.79	1242.76	924.93
	4	ELEMENTS	368.11	92.93	31.67
		CHOLESKY	2500.95	851.43	715.59
		Total	2949.26	997.46	780.61

**Fig. 24.7** Scalability of B.2P.DZ.pqgt1t2 w.r.t. the number of nodes (left) and threads (right)

linear scalability. The row-wise distribution plays a crucial role in this case. It does not require any network communication and makes all nodes concentrate on the evaluations of the assigned rows. It is also effective when the number of threads goes from 1 to 4 on 1 node; the scalability is surprisingly 4.0. The first-come-first-assigned row-wise distribution successfully manages all threads and improves scalability. Consequently, the hybrid row-wise distribution scores 62.28 on  $16 \times 4 = 64$  threads. Since the bulk of the computation time of B.2P.DZ.pqgt1t2 is occupied by the ELEMENTS component, this scalability leads a total scalability of 45.86 on 64 threads. Hence, we can reduce the computation time from 8.25 h to 11 min.

The scalability of optControl(20,8,5,0) is also good. Figure 24.8 plots the scalability of optControl(20,8,5,0) versus the number of nodes (left figure)



**Fig. 24.8** Scalability of  $\text{optControl}(20,8,5,0)$  w.r.t. the number of nodes (*left*) and threads (*right*)

and threads (right figure). Note that the SCM of the parallel evaluation in  $\text{optControl}(20,8,5,0)$  is sparse. These figures indicate that the parallel evaluation works well for the ELEMENTS component for the sparse SCM. More specifically, the scalability is 35.91 for 64 threads. Although this scalability is lower than that of B.2P.DZ.pqgt1t2, it is still good because of the elaborate management for the sparse SCM. In contrast to the ELEMENTS component, it is difficult to attain good scalability for the CHOLESKY component for a sparse SCM. This lowers the total scalability of SDPARA for sparse SCMs. The parallel sparse Cholesky factorization of MUMPS is more difficult to exploit effectively in comparison with dense Cholesky factorization. In fact, we chose MUMPS because no other software can attain sufficient scalability in parallel environments with more than four nodes. The scalability for the CHOLESKY component obtained by the parallel Cholesky factorization is 5.66 on 16 nodes and 4 threads. Although it is smaller than the scalability for the dense SCM, the total scalability is still good (6.71 on 16 nodes with 4 threads).

Table 24.3 summarizes the times taken to solve other SDPs using 1, 2 or 4 threads. The best result is shown in the corresponding row. For example, the best result for “control11” is obtained with 4 threads. Note that more threads do not always reduce the computation time. In nonscomp(30,000), we have 90,000 sub-matrices in  $\mathbf{X}$  and  $\mathbf{Y}$  and the size of each sub-matrix is just  $3 \times 3$  or  $6 \times 6$ . In this case, the computation bottleneck becomes the multiplications of  $d\mathbf{XY}$  and  $\mathbf{X}^{-1}(\mathbf{R} - d\mathbf{XY})$  in Step 5 of the algorithmic framework. In this step, we invoke 90,000 multi-threaded multiplications of  $3 \times 3$  or  $6 \times 6$  dimensions. For such SDPs, a single thread is superior to multiple threads.

The SDPs from SDPLIB have fully-dense SCMs and fewer sub-matrices, and SDPARA solves them faster and attains high scalability. For example, its scalability on control11 is 12.7 from 1 node/1 thread (94.56 s) to 16 nodes/4 threads (7.40 s). In particular, the ELEMENTS component scores 32.3.

The SDPs of POP are not suitable for efficient computation in SDPARA since they have a large number of sub-matrices. In addition, it is difficult to obtain accurate

**Table 24.3** Computation time of SDPARA (in seconds)

Name	#Threads	#Nodes	1	4	16
control11	4	ELEMENTS	30.58	8.15	2.65
		CHOLESKY	4.05	2.58	2.16
		Total	40.15	14.56	7.40
thetaG51	4	ELEMENTS	32.13	9.20	2.87
		CHOLESKY	184.76	40.65	20.59
		Total	225.14	83.06	45.50
optControl(20,8,6,0)	4	ELEMENTS	760.07	190.11	86.09
		CHOLESKY	4916.53	1892.91	1754.01
		Total	5818.22	2173.92	1901.03
nonscomp(30,000)	1	ELEMENTS	13.66	9.66	6.84
		CHOLESKY	20.33	17.48	15.92
		Total	88.99	89.77	82.36
		#Nodes		8	16
CH2.3B1.DZ.pqgt1t2p	4	ELEMENTS		57211.09	28678.10
		CHOLESKY		1326.31	745.47
		Total		61573.50	32275.63

solutions for these SDPs. In spite of these difficulties, SDPARA still attains high scalability. For these applications, we should carefully select the numbers of nodes and threads.

The last point on which we want to remark is that CH2.3B1.DZ.pqgt1t2 and optControl(20,8,6,0) are extremely large-scale SDPs. In particular, a single node could not solve CH2.3B1.DZ.pqgt1t2 because of lack of memory. At present, SDPARA is the only software package that can solve these SDPs. Furthermore, the scalability of SDPARA usually improves when the SDPs become larger.

## 24.5 Some Remarks and Future Work

In this chapter, we outlined the software of the SDPA Family and described numerical results on SDPARA. Each software package has its own capabilities. The numerical results verify SDPARA's ability to solve large-scale SDPs. We hope that some readers will consider using the SDPA Family to solve their own SDPs. The SDPA Online Solver will be a good starting point for them.

We are continuing to improve the performance of the SDPA Family. The numerical experiments presented here point out some aspects for improvement.

- The performance is poor when the diagonal block structure has too many small sub-matrices. The sub-matrices could be combined into a larger sub-matrix by using appropriate thresholds.
- Multiple threads computation is very efficient. However, there are resource conflicts between multiple threads for some SDPs and they lower ELEMENTS component's scalability. A variant of CSDP for the shared memory architec-

ture [9] faces similar problems, such as excessive use of memory space because of simultaneous accesses from all threads. The variant uses the multiple threads indirectly via the OpenMP framework; hence, it is difficult to control each thread. Since SDPA and SDPARA manipulates the threads directly, we believe that we can significantly improve their performance in multiple threads computations.

**Acknowledgements** M. Yamashita's and K. Nakata's research was partially supported by Grant-in-Aid for Young Scientists (B) 21710148 and (B) 22710136. K. Fujisawa's, M. Fukuda's and M. Nakata's research was partially supported by Grant-in-Aid for Scientific Research (C) 20510143, (B) 20340024, and (B) 21300017. K. Fujisawa's research was also supported by a Chuo University Grant for Special Research.

## References

1. Alizadeh, F., Haeberly, J.-P.A., Overton, M.L.: Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results. *SIAM J. on Optim.* **8**, 746–768 (1998)
2. Amestoy, P.R., Duff, I.S., L'Excellent, J.-Y.: Multifrontal parallel distributed symmetric and unsymmetric solvers. *Comput. Methods in Appl. Mech. Eng.* **184**, 501–520 (2000)
3. Amestoy, P.R., Duff, I.S., L'Excellent, J.-Y., Koster, J.: A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM J. of Matrix Anal. and Appl.* **23**, 15–41 (2001)
4. Amestoy, P.R., Duff, I.S., L'Excellent, J.-Y., Pralet, S.: Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing* **32**, 136–156 (2006)
5. Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network localization. In: Proceedings of the third international symposium on information processing in sensor networks, ACM press, 46–54 (2004)
6. Blackford, L.S., Choi, J., Cleary, A., D'Azevedo, E., Demmel, J., Dhillon, I., Dongarra, J., Hammarling, S., Henry, G., Petitet, A., Stanley, K., Walker, D., Whaley, R.C.: *ScalAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia (1997)
7. Borchers, B.: CSDP, A C library for semidefinite programming. *Optim. Meth. and Softw.* **11 & 12**, 613–623 (1999)
8. Borchers, B.: SDPLIB 1.2, a library of semidefinite programming test problems. *Optim. Meth. and Softw.* **11 & 12**, 683–690 (1999)
9. Borchers, B., Young, J.G.: Implementation of a primal-dual method for SDP on a shared memory parallel architecture. *Comp. Optim. and Appl.* **37**, 355–369 (2007)
10. Boyd, S., Ghaoui, L.E., Feron, E., Balakrishnan, V.: *Linear matrix inequalities in system and control theory*. Society for Industrial and Applied Mathematics, Philadelphia, PA (1994)
11. Choi, J., Dongarra, J., Ostrouchov, S., Petitet, A., Walker, D., Whaley, R.C.: The design and implementation of the ScalAPACK LU, QR, and Cholesky factorization routines. Tech. Report UT CS-94-296, LAPACK Working Notes #80, University of Tennessee (1994)
12. Fujisawa, K., Kojima, M., Nakata, K.: Exploiting sparsity in primal-dual interior-point methods for semidefinite programming. *Math. Prog.* **79**, 235–253 (1997)
13. Fujisawa, K., Nakata, K., Yamashita, M., Fukuda, M.: SDPA Project: Solving large-scale semidefinite programs. *J. Oper. Res. Soc. Japan* **50**, 278–298 (2007)
14. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.* **42**, 1115–1145 (1995)

15. Grone, R., Johnson, C.R., Sá, E.M., Wolkowicz, H.: Positive definite completions of partial Hermitian matrices, *Linear Algebra Appl.* **58**, 109–124 (1984)
16. Helberg, C., Rendl, F., Vanderbei R.J., Wolkowicz, H.: An interior-point method for semidefinite programming. *SIAM J. Optim.* **6**, 342–361 (1996)
17. Hida, Y., Li, X.S., Bailey, D.H.: Quad-double arithmetic: Algorithms, implementation, and application, Technical Report LBNL-46996, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, Oct. (2000)
18. Kobayashi, K., Kim S., Kojima, M.: Correlative sparsity in primal-dual interior-point methods for LP, SDP and SOCP, *Appl. Math. Optim.* **58**, 69–88 (2008)
19. Kojima, M., Shindoh, S., Hara, S.: Interior-point methods for the monotone semidefinite linear complementarity problems in symmetric matrices. *SIAM J. Optim.* **7**, 86–125 (1997)
20. Lasserre, J.B.: Global optimization with polynomials and the problems of moments. *SIAM J. Optim.* **11**, 796–817 (2001)
21. Liu, Y.T., Liu, T.Y., Qin, T., Ma, Z.M., Li, H.: Supervised rank aggregation. Proceedings of the 16th international conference on World Wide Web, 481–490 (2007)
22. Löfberg, J.: YALMIP : A toolbox for modeling and optimization in MATLAB. Proceedings of the CACSD Conference (2004)
23. Mittelman, H.D.: Additional SDP test problems. <http://plato.asu.edu/ftp/sdp/> 00README
24. Mittelmann, H.D., Vallentin, F.: High accuracy semidefinite programming bounds for kissing numbers. *Exper. Math.* **19**, 174–179 (2010)
25. Monteiro, R.D.C.: Primal-dual path following algorithms for semidefinite programming. *SIAM J. Optim.* **7**, 663–678 (1997)
26. Nakata, K., Fujisawa, K., Fukuda, M., Kojima, M., Murota, K.: Exploiting sparsity in semidefinite programming via matrix completion II: Implementation and numerical results, *Math. Prog. B* **95**, 303–327 (2003)
27. Nakata, K., Yamashita, M., Fujisawa, K., Kojima, M.: A parallel primal-dual interior-point method for semidefinite programs using positive definite matrix completion, *Parallel Computing* **32**, 24–43 (2006).
28. Nakata, M., Braams, B.J., Fujisawa, K., Fukuda, M., Percus, J.K., Yamashita, M., Zhao, Z.: Variational calculation of second-order reduced density matrices by strong  $N$ -representability conditions and an accurate semidefinite programming solver. *J. Chem. Phys.* **128**, 164113 (2008)
29. Nakata, M., Nakatsuji, H., Ehara, M., Fukuda, M., Nakata, K., Fujisawa, K.: Variational calculations of fermion second-order reduced density matrices by semidefinite programming algorithm. *J. Chem. Phys.* **114**, 8282–8292 (2001)
30. Nesterov, Y.E., Todd, M.J.: Primal-dual interior-point methods for self-scaled cones. *SIAM J. Optim.* **8**, 324–364 (1998)
31. Sturm, J.F.: SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Meth. and Softw.* **11 & 12**, 625–653 (1999)
32. Toh, K.-C., Todd, M.J., Tütüncü, R.H.: SDPT3 – a MATLAB software package for semidefinite programming, version 1.3. *Optim. Meth. and Softw.* **11 & 12**, 545–581 (1999)
33. Waki, H., Kim, S., Kojima, M., Muramatsu, M., Sugimoto, H.: Algorithm 883: SparsePOP : A Sparse semidefinite programming relaxation of Polynomial Optimization Problems. *ACM Trans. Math. Softw.* **35**, 13 pages (2009)
34. Waki, H., Nakata, M., Muramatsu, M.: Strange behaviors of interior-point methods for solving semidefinite programming problems in polynomial optimization. To appear in *Comput. Optim. and Appl.*
35. Yamashita, M., Fujisawa, K., Fukuda, M., Nakata, K., Nakata, M.: Parallel solver for semidefinite programming problems having sparse Schur complement matrix. Research Report B-463, Dept. of Math. and Comp. Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152–8552, September 2010.
36. Yamashita, M., Fujisawa, K., Kojima, M.: Implementation and evaluation of SDPA6.0 (SemiDefinite Programming Algorithm 6.0). *Optim. Meth. and Softw.* **18**, 491–505 (2003)

37. Yamashita, M., Fujisawa, K., Kojima, M.: SDPARA: SemiDefinite Programming Algorithm paRALlel version. *Parallel Computing* **29**, 1053–1067 (2003)
38. Yamashita, M., Fujisawa, K., Nakata, K., Nakata, M., Fukuda, M., Kobayashi, K., Goto, K.: A high-performance software package for semidefinite programs: SDPA 7. Research Report B-460, Dept. of Math. and Comp. Sciences, Tokyo Institute of Technology, Oh-Okayama, Meguro, Tokyo 152–8552, January 2010.

# Chapter 25

## On the Implementation and Usage of SDPT3 – A MATLAB Software Package for Semidefinite-Quadratic-Linear Programming, Version 4.0

Kim-Chuan Toh, Michael J. Todd, and Reha H. Tütüncü

### 25.1 Introduction

The current version of SDPT3, version 4.0, is designed to solve conic programming problems whose constraint cone is a product of semidefinite cones, second-order cones, nonnegative orthants and Euclidean spaces, and whose objective function is the sum of linear functions and log-barrier terms associated with the constraint cones. It solves the following standard form of such problems, henceforth called standard SQLP problems:

$$(P) \min \sum_{j=1}^{n_s} [\langle c_j^s, x_j^s \rangle - v_j^s \log \det(x_j^s)] + \sum_{i=1}^{n_q} [\langle c_i^q, x_i^q \rangle - v_i^q \log \gamma(x_i^q)] \\ + \langle c^l, x^l \rangle - \sum_{k=1}^{n_l} v_k^l \log x_k^l + \langle c^u, x^u \rangle \\ \text{s.t. } \sum_{j=1}^{n_s} \mathcal{A}_j^s(x_j^s) + \sum_{i=1}^{n_q} A_i^q x_i^q + A^l x^l + A^u x^u = b, \\ x_j^s \in K_s^{s_j} \forall j, x_i^q \in K_q^{q_i} \forall i, x^l \in K_l^{n_l}, x^u \in R^{n_u}.$$

---

K.-C. Toh (✉)

Department of Mathematics, National University of Singapore, Blk S17,  
10 Lower Kent Ridge Road, Singapore 119076  
e-mail: [mattohkc@nus.edu.sg](mailto:mattohkc@nus.edu.sg)

M.J. Todd

School of Operations Research and Information Engineering,  
Cornell University, Ithaca, NY 14853, USA  
e-mail: [miketodd@cs.cornell.edu](mailto:miketodd@cs.cornell.edu)

R.H. Tütüncü

Quantitative Investment Strategies, Goldman Sachs Asset Management,  
New York, NY 10282, USA  
e-mail: [reha.tutuncu@goldmansachs.com](mailto:reha.tutuncu@goldmansachs.com)

Here,  $c_j^s, x_j^s$  lie in the space  $\mathcal{S}^{s_j}$  of real symmetric matrices of order  $s_j$  and  $K_s^{s_j}$  is the cone of positive semidefinite symmetric matrices of the same order. Similarly,  $c_i^q, x_i^q$  are vectors in  $\mathbf{R}^{q_i}$  and  $K_q^{q_i}$  is the quadratic or second-order cone defined by  $K_q^{q_i} := \{x = [x_0; \bar{x}] \in \mathbf{R}^{q_i} : x_0 \geq \sqrt{\bar{x}^T \bar{x}}\}$ . Finally,  $c^l, x^l$  are real vectors of dimension  $n_l$ ,  $K_l^{n_l}$  is the nonnegative orthant  $\mathbf{R}_+^{n_l}$ , and  $c^u, x^u$  are real vectors of dimension  $n_u$ . In the notation above,  $\mathcal{A}_j^s$  is the linear map from  $K_s^{s_j}$  to  $\mathbf{R}^m$  defined by

$$\mathcal{A}_j^s(x_j^s) = [\langle a_{j,1}^s, x_j^s \rangle; \dots; \langle a_{j,m}^s, x_j^s \rangle],$$

where  $a_{j,1}^s, \dots, a_{j,m}^s \in \mathcal{S}^{s_j}$  are constraint matrices associated with the  $j$ th semidefinite block variable  $x_j^s$ . The matrix  $A_i^q$  is an  $m \times q_i$  dimensional constraint matrix corresponding to the  $i$ th quadratic block variable  $x_i^q$ , and  $A^l$  and  $A^u$  are the  $m \times n_l$  and  $m \times n_u$  dimensional constraint matrices corresponding to the linear block variable  $x^l$  and the unrestricted block variable  $x^u$ , respectively. The notation  $\langle p, q \rangle$  denotes the standard inner product in the appropriate space. For a given vector  $u = [u_0; \bar{u}]$  in a second-order cone, we define  $\gamma(u) := \sqrt{u_0^2 - \bar{u}^T \bar{u}}$ . In the problem  $(P)$ ,  $v_j^s, v_i^q$ , and  $v_k^l$  are given nonnegative parameters.

In this chapter, the vector 2-norm and matrix Frobenius norm are denoted by  $\|\cdot\|$ . We use the MATLAB notation  $[U; V]$  to denote the matrix obtained by appending  $V$  below the last row of  $U$ . For a given matrix  $U$ , we use the notation  $U(k, \cdot)$  and  $U(\cdot, k)$  to denote the  $k$ th row and column of  $U$ , respectively.

The software also solves the dual problem associated with the problem  $(P)$  above:

$$(D) \max b^T y + \sum_{j=1}^{n_s} [v_j^s \log \det(z_j^s) + s_j v_j^s (1 - \log v_j^s)] \\ + \sum_{i=1}^{n_q} [v_i^q \log \gamma(z_i^q) + v_i^q (1 - \log v_i^q)] \\ + \sum_{k=1}^{n_l} [v_k^l \log z_k^l + v_k^l (1 - \log v_k^l)]$$

s.t.  $(\mathcal{A}_j^s)^T y + z_j^s = c_j^s, \quad z_j^s \in K_s^{s_j}, \quad j = 1 \dots, n_s$   
 $(A_i^q)^T y + z_i^q = c_i^q, \quad z_i^q \in K_q^{q_i}, \quad i = 1 \dots, n_q$   
 $(A^l)^T y + z^l = c^l, \quad z^l \in K_l^{n_l}, \quad (A^u)^T y = c^l, \quad y \in \mathbf{R}^m.$

In the notation above,  $(\mathcal{A}_j^s)^T$  is the adjoint of  $\mathcal{A}_j^s$  defined by  $(\mathcal{A}_j^s)^T y = \sum_{k=1}^m y_k a_{j,k}^s$ .

Let  $\text{svec} : \mathcal{S}^n \rightarrow \mathbf{R}^{n(n+1)/2}$  be the vectorization operator on symmetric matrices defined by  $\text{svec}(X) = [X_{11}, \sqrt{2}X_{12}, X_{22}, \dots, \sqrt{2}X_{1n}, \dots, \sqrt{2}X_{n-1,n}, X_{nn}]^T$ . For computational purposes, it is convenient to identify  $\mathcal{A}_j^s$  with the following  $m \times \bar{s}_j$  matrix (where  $\bar{s}_j = s_j(s_j + 1)/2$ ):

$$A_j^s = [\text{svec}(a_{j,1}^s); \dots; \text{svec}(a_{j,m}^s)].$$

With the matrix representation of  $\mathcal{A}_j^s$ , we have that  $\mathcal{A}_j^s(x_j^s) = A_j^s \text{svec}(x_j^s)$ .

For later convenience, we introduce the following notation:

$$x^s = (x_1^s; \dots; x_{n_s}^s), \quad x^q = [x_1^q; \dots; x_{n_q}^q], \quad A^q = [A_1^q; \dots; A_{n_q}^q],$$

where the notation  $(x_1^s; \dots; x_{n_s}^s)$  means that the objects  $x_j^s$  are placed in a column format. We define  $c^s, z^s, c^q$ , and  $z^q$  analogously. Let  $\mathcal{A}^s(x^s) = \sum_{j=1}^{n_s} \mathcal{A}_j^s(x_j^s)$ ,  $(\mathcal{A}^s)^T y = ((\mathcal{A}_1^s)^T y; \dots; (\mathcal{A}_{n_s}^s)^T y)$ , and  $c = (c^s; c^q; c^l; c^u)$ ,  $x = (x^s; x^q; x^l; x^u)$ ,  $z = (z^s; z^q; z^l; 0)$ . Finally, we define

$$\mathcal{A}(x) = \mathcal{A}^s(x^s) + A^q x^q + A^l x^l + A^u x^u,$$

$$\mathcal{A}^T(y) = ((\mathcal{A}^s)^T y; (A^q)^T y; (A^l)^T y; (A^u)^T y),$$

$$K = K_s^{s_1} \times \dots \times K_s^{s_{n_s}} \times K_q^{q_1} \times \dots \times K_q^{q_{n_q}} \times K_l^{n_l} \times R^{n_u},$$

$$K^* = K_s^{s_1} \times \dots \times K_s^{s_{n_s}} \times K_q^{q_1} \times \dots \times K_q^{q_{n_q}} \times K_l^{n_l} \times \{0\},$$

so that the equality constraints of  $(P)$  and  $(D)$  can be written more compactly as

$$\mathcal{A}(x) = b, \quad x \in K, \quad \mathcal{A}^T(y) + z = c, \quad z \in K^*. \quad (25.1)$$

The matrix representation of  $\mathcal{A}$  is given by

$$A = [A_1^s; \dots; A_{n_s}^s, A^q, A^l, A^u]. \quad (25.2)$$

The software package was originally developed to provide researchers in semidefinite programming with a collection of reasonably efficient and robust algorithms that could solve general SDPs (semidefinite programming problems) with matrices of dimensions of the order of a hundred. The current release expands the family of problems solvable by the software and made several enhancements described below.

1. This version is faster than the previous releases, e.g. [31, 33], especially on large sparse problems, and consequently can solve larger problems.
2. The current release can also solve problems that have explicit log-barrier terms in the objective functions. Hence determinant maximization problems can be solved.
3. The solver is more robust in handling unrestricted variables.
4. Low-rank structures in the constraint matrices associated with the semi-definite blocks can be exploited to improve computational efficiency and memory requirements.
5. The current release also implements primal-dual predictor-corrector path-following algorithms for solving a 3-parameter homogeneous self-dual model of  $(P)$  and  $(D)$ .

6. Routines are provided to compute the geometric measures proposed by Freund [8] for  $(P)$  and  $(D)$ . These geometric measures give information on the “thickness” of the feasible regions of  $(P)$  and  $(D)$ .

All the numerical experiments in this chapter were performed on an Intel Xeon 3.0 GHz personal computer with 4 GB of physical memory using MATLAB version 7.6 on a Linux operating system.

### ***25.1.1 Organization of the Chapter***

In Sect. 25.2, we describe the representation of SQLP data by cell arrays. The SQLP solver `sqlp.m` in our software is described in Sect. 25.3. In Sect. 25.4, we present a few SQLP examples to illustrate the usage of our software. Implementation details such as the computation of search directions are given in Sect. 25.5. In Sect. 25.6, we describe a 3-parameter homogeneous self-dual model for the problem  $(P)$  without logarithmic terms and unrestricted variables. In Sect. 25.7, the geometric measures of Freund and their computation are presented. Finally, the last section reports computational results obtained by SDPT3 on about 430 SQLP problems.

### ***25.1.2 Installation***

The current version is written in MATLAB version 7.4 or later releases. It is available from the web site: <http://www.math.nus.edu.sg/~mattohkc/sdpt3.html>. Our software uses a number of Mex routines generated from C programs written to carry out certain operations that MATLAB is not efficient at. In particular, operations such as extracting selected elements of a matrix, and performing arithmetic operations on these selected elements are all done in C. As an example, the vectorization operation `svec` is coded in the C program `mexsvec.c`. To install SDPT3 and generate these Mex routines, the user can simply follow the instructions given in the above web site.

## **25.2 Cell Array Representation of Problem Data**

Our implementation exploits the block structure of the given SQLP problem. In the internal representation of the problem data, we classify each semidefinite block into one of the following two types:

1. A dense or sparse matrix of order greater than or equal to 100.
2. A sparse block-diagonal matrix consisting of numerous sub-blocks each of order less than 100.

The reason for using the sparse matrix representation to handle the case when we have numerous small diagonal blocks is that it is less efficient for MATLAB to work with a large number of cell array elements compared to working with a single cell array element consisting of a large sparse block-diagonal matrix. Technically, no problem will arise if one chooses to store the small blocks individually instead of grouping them together as a sparse block-diagonal matrix.

For the quadratic part, we typically group all quadratic blocks (small or large) into a single block, though it is not mandatory to do so. If there are a large number of small blocks, it is advisable to group them all together as a single large block consisting of numerous small sub-blocks for the same reason we just mentioned.

Let  $L$  be the total number of blocks in the SQLP problem. If all the various types of blocks are present in  $(P)$ , then  $L = n_s + n_q + 2$ . For each SQLP problem, the block structure of the problem data is described by an  $L \times 2$  cell array named `blk`. The content of each of the elements of the cell arrays is given as follows. If the  $j$ th block is a semidefinite block consisting of a single block of size  $s_j$ , then

$$\text{blk}\{j, 1\} = 's', \quad \text{blk}\{j, 2\} = [s_j]$$

$$\text{At}\{j\} = [\bar{s}_j \times m \text{ sparse}], \quad C\{j\}, X\{j\}, Z\{j\} = [s_j \times s_j \text{ double or sparse}],$$

where  $\bar{s}_j = s_j(s_j + 1)/2$ .

If the  $j$ th block is a semidefinite block consisting of numerous small sub-blocks, say  $p$  of them, of orders  $s_{j1}, s_{j2}, \dots, s_{jp}$  such that  $\sum_{k=1}^p s_{jk} = s_j$ , then

$$\text{blk}\{j, 1\} = 's', \quad \text{blk}\{j, 2\} = [s_{j1}, s_{j2}, \dots, s_{jp}]$$

$$\text{At}\{j\} = [\bar{s}_j \times m \text{ sparse}], \quad C\{j\}, X\{j\}, Z\{j\} = [s_j \times s_j \text{ sparse}],$$

where  $\bar{s}_j = \sum_{k=1}^p s_{jk}(s_{jk} + 1)/2$ .

Notice that we store all the constraint matrices associated with the  $j$ th semidefinite block in vectorized form as a single  $\bar{s}_j \times m$  matrix where the  $k$ th column of this matrix corresponds to the  $k$ th constraint matrix. That is,  $\text{At}\{j\}(:, k) = \text{svec}(\text{blk}\{j, :, k\})$ . (Here `svec` has a new argument because, if the  $j$ th semidefinite block consists of several small sub-blocks, it needs to concatenate the `svec`'s of each sub-block.)

The above storage scheme for the data matrix  $A_j^s$  associated with the semidefinite blocks of the SQLP problem represents a departure from earlier versions (version 2.3 or earlier) of our implementation, such as the one described in [31]. Previously, the constraint matrices were stored in an  $n_s \times m$  cell array `AA`, where  $\text{AA}\{j, k\} = a_{j,k}^s$ , and it was stored as an individual matrix in either dense or sparse format. The data format we used in earlier versions of SDPT3 was more natural, but our current data representation was adopted for the sake of computational efficiency. The reason for such a change is again due to the fact that it is less efficient for MATLAB to work with a cell array with many cells. But note that it is easy to use the function `svec.m` provided in SDPT3 to convert `AA` into the new storage scheme as follows:  $\text{At}(j) = \text{svec}(\text{blk}(j, :, \text{AA}(j, :)))$ .

While we now store the constraint matrix in vectorized form, the parts of the iterates  $X$  and  $Z$  corresponding to semidefinite blocks are still stored as matrices, since that is how the user wants to access them.

The data storage scheme corresponding to quadratic, linear, and unrestricted blocks is rather straightforward. If the  $i$ th block is a quadratic block consisting of numerous sub-blocks, say  $p$  of them, of dimensions  $q_{i1}, q_{i2}, \dots, q_{ip}$  such that  $\sum_{k=1}^p q_{ik} = q_i$ , then

$$\text{blk}\{i, 1\} = 'q', \quad \text{blk}\{i, 2\} = [q_{i1}, q_{i2}, \dots, q_{ip}]$$

$$\text{At}\{i\} = [q_i \text{ xm sparse}], \quad C\{i\}, X\{i\}, Z\{i\} = [q_i \text{ x 1 double or sparse}].$$

If the  $k$ th block is the linear block, then

$$\text{blk}\{k, 1\} = 'l', \quad \text{blk}\{k, 2\} = n_l$$

$$\text{At}\{k\} = [n_l \text{ xm sparse}], \quad C\{k\}, X\{k\}, Z\{k\} = [n_l \text{ x 1 double or sparse}].$$

Similarly, if the  $k$ th block is the unrestricted block, then

$$\text{blk}\{k, 1\} = 'u', \quad \text{blk}\{k, 2\} = n_u$$

$$\text{At}\{k\} = [n_u \text{ xm sparse}], \quad C\{k\}, X\{k\}, Z\{k\} = [n_u \text{ x 1 double or sparse}].$$

(It is possible to have several linear or unrestricted blocks, but it is more efficient to reformulate such a problem by combining all linear blocks and similarly all unrestricted blocks.)

### 25.2.1 Specifying the Block Structure of Problems

Our software requires the user to specify the block structure of the SQLP problem. Although no technical difficulty will arise if the user chooses to lump a few blocks together and consider it as a single large block, the computational time can be dramatically different. For example, the problem qpG11 in the SDPLIB library [2] actually has the block structure:  $\text{blk}\{1, 1\} = 's'$ ,  $\text{blk}\{1, 2\} = 800$ ,  $\text{blk}\{2, 1\} = 'l'$ ,  $\text{blk}\{2, 2\} = 800$ , but the structure specified in the library is  $\text{blk}\{1, 1\} = 's'$ ,  $\text{blk}\{1, 2\} = 1,600$ . That is, in the former, the linear variables are explicitly identified, rather than being part of a large sparse semidefinite block. The difference in the running time for specifying the block structure differently is dramatic: the former representation is at least six times faster when the HKM direction is used, besides using much less memory space.

It is thus crucial to present problems to the algorithms correctly. We could add our own preprocessor to detect this structure, but believe users are aware of linear variables present in their problems. Unfortunately the versions of qpG11 (and also

`qpG51`) in SDPLIB do not show this structure explicitly. In our software, we provide an m-file, `detect_lblk.m`, to detect problems with linear variables. The user can call this m-file after loading the problem data into MATLAB as follows:

```
>> [blk,At,C,b] = read_sdpa('./sdplib/qpG11.dat-s');
>> [blk2,At2,C2] = detect_lblk(blk,At,C,b);
```

Internally, the solvers in SDPT3 would automatically call `detect_lblk.m` to detect the presence of linear variables in a semidefinite block before solving  $(P)$  and  $(D)$ .

### 25.2.2 Storing Constraint Matrices with Low-Rank Structures

A new feature of the current version of SDPT3 is that it can exploit low-rank structures present in the constraint matrices associated with the semidefinite blocks. To do so, the user needs to specify the low-rank structures in the constraint matrices explicitly when coding the problem data. The purpose here is to explain how this is done.

Suppose the  $j$ th row of `blk` corresponds to a semidefinite block. To simplify implementation, we exploit possible low-rank structures only when this semidefinite block is a single block. That is,  $\text{blk}\{j,2\} = [\mathbf{s}_j]$ . Suppose that the first  $p$  matrices,  $a_{j,1}^s, \dots, a_{j,p}^s$ , have no low-rank structures, and the remaining matrices  $a_{j,p+1}^s, \dots, a_{j,m}^s$  have such structures with

$$a_{j,k}^s = V_k D_k V_k^T, \quad k = p + 1, \dots, m,$$

where  $V_k \in \mathbb{R}^{s_j \times r_{j,k}}$  is a low-rank matrix with  $r_{j,k} \ll s_j$ , and  $D_k \in \mathbb{R}^{r_{j,k} \times r_{j,k}}$  is a symmetric matrix. The low-rank structures of these matrices should be recorded as follows:

$$\begin{aligned} \text{blk}\{j,1\} &= 's', \quad \text{blk}\{j,2\} = [\mathbf{s}_j], \quad \text{blk}\{j,3\} = [\mathbf{r}_{j,p+1}, \dots, \mathbf{r}_{j,m}], \\ \text{At}\{j,1\} &= [\bar{\mathbf{s}}_j \times \mathbf{p} \text{ sparse}], \quad \text{At}\{j,2\} = [\mathbf{V}_{j,p+1}, \dots, \mathbf{V}_{j,m}], \quad \text{At}\{j,3\} = \mathbf{dd}, \end{aligned}$$

where  $\mathbf{dd}$  is a 4-column matrix that records the non-zero elements of  $D_k$ ,  $k = p + 1, \dots, m$ , and a row (say,  $i$ th row) of  $\mathbf{dd}$  has the following form:

$$\mathbf{d}(i,:) = [\text{constraint number} - p, \text{row index}, \text{column index}, \text{non-zero value}].$$

If all the matrices  $D_k$  are diagonal, then one can simply set  $\mathbf{dd}$  as follows:

$$\mathbf{dd} = [\text{diag}(D_{p+1}); \dots; \text{diag}(D_m)].$$

In the subdirectory `Examples`, we give an m-file `randlowranksdp.m` to generate random SDP problems with low-rank constraint matrices, whose calling syntax is:

```
[blk,At,C,b,bblk,AAt] = randlowranksdp(n,p,m2,r)
```

It will generate an SDP where the first  $p$  constraint matrices have no low-rank structures, and the remaining  $m2$  matrices have low-rank structures and each matrix has rank  $r$ . The output `[blk,At,C,b]` explicitly describes the low-rank structure as above, while `[bblk,AAt,C,b]` encodes the same SDP, but without including the low-rank structure information.

### 25.3 The Main Functions: `sqlp.m`, `HSDsqlp.m`, `sdpt3.m`

The main algorithm implemented in SDPT3 for solving  $(P)$  and  $(D)$  is an infeasible primal-dual path-following algorithm, described in detail in Appendix A. At each iteration, we first compute a *predictor* search direction aimed at decreasing the duality gap as much as possible. After that, the algorithm generates a Mehrotra-type corrector step [20] with the intention of keeping the iterates close to the central path. However, we do not impose any neighborhood restrictions on our iterates.<sup>1</sup> Initial iterates need not be feasible – the algorithm tries to achieve feasibility and optimality of its iterates simultaneously. It should be noted that in our implementation, the user has the option to use a primal-dual path-following algorithm that does not use corrector steps.

The main routine that corresponds to Algorithm IPC described in Appendix A for solving  $(P)$  and  $(D)$  is `sqlp.m`, whose calling syntax is as follows:

```
[obj,X,y,Z,info,runhist] = sqlp(blk,At,C,b,OPTIONS,X0,y0,Z0).
```

For an SQLP problem without logarithmic terms or unrestricted variables  $x^u$ , we also implemented an algorithm that is analogous to Algorithm IPC for a 3-parameter homogeneous self-dual model of  $(P)$ . The routine for solving the HSD model of  $(P)$  is `HSDsqlp.m`, whose calling syntax is similar to that of `sqlp.m`. Note that if there are unrestricted variables  $x^u$  present in  $(P)$ , `HSDsqlp.m` can still be called to solve the problem since it would automatically express  $x^u$  as  $x^u = x_+^u - x_-^u$  with  $x_+^u, x_-^u \geq 0$  to reformulate  $(P)$  into an SQLP without unrestricted variables. Our

---

<sup>1</sup>This strategy works well on most of the problems we tested. However, it should be noted that the occasional failure of the software on problems with poorly chosen initial iterates is likely due to the lack of a neighborhood enforcement in the algorithm.

numerical experience has indicated that for an SQLP with unrestricted variables in  $(P)$ , `HSDsqlp.m` would typically deliver more accurate solutions than `sqlp.m` since the former generally would encounter less severe numerical difficulties compared to the latter. But for an SQLP problem without unrestricted variables, `HSDsqlp.m` generally would take more iterations than `sqlp.m` to solve the problem to the same accuracy. Based on the consideration of computational efficiency and attainable accuracy/robustness, we have a hybrid solver `sdpt3.m` that would automatically choose between `sqlp.m` and `HSDsqlp.m` based on the characteristics of the SQLP problem to be solved. Again, the calling syntax of `sdpt3.m` is similar to that for `sqlp.m`.

### 25.3.1 Input Arguments

`blk`: a cell array describing the block structure of the SQLP problem.

`At, C, b`: SQLP data.

`OPTIONS`: a structure array of parameters (optional).

`X0, y0, Z0`: an initial iterate (optional).

If the input argument `OPTIONS` is omitted, default values specified in the function `sqlpparameters.m` are used. More detail on `OPTIONS` is given in Sect. 25.3.4.

### 25.3.2 Output Arguments

The names chosen for the output arguments explain their contents. The argument `info` is a structure array containing performance information such as `info.termcode`, `info.obj`, `info.gap`, `info.pinfeas`, `info.dinfeas`, `info.cputime` whose meanings are explained in `sqlp.m`. The argument `runhist` is a structure array which records the history of various performance measures during the run; for example, `runhist.gap` records the complementarity gap at each interior-point iteration.

Note that, while  $(X, y, Z)$  normally gives approximately optimal solutions, if `info.termcode` is 1 the problem is suspected to be primal infeasible and  $(y, Z)$  is an approximate certificate of infeasibility, with  $b^T y = 1$ ,  $Z$  in the appropriate cone, and  $A^T y + Z$  small, while if `info.termcode` is 2 the problem is suspected to be dual infeasible and  $X$  is an approximate certificate of infeasibility, with  $\langle C, X \rangle = -1$ ,  $X$  in the appropriate cone, and  $AX$  small. Note that  $A$  is defined in (25.2).

### 25.3.3 *Caveats*

- (a) The user should be aware that SQLP is more complicated than linear programming. For example, it is possible that both primal and dual problems are feasible, but their optimal values are not equal. Also, either problem may be infeasible without there being a certificate of that fact (so-called weak infeasibility). In such cases, our software package is likely to terminate after some iterations with an indication of short step-length or lack of progress. Also, even if there is a certificate of infeasibility, our infeasible-interior-point methods may not find it. In our very limited testing on strongly infeasible problems, our algorithms have been quite successful in detecting infeasibility.
- (b) Since our algorithm is a primal-dual method storing the primal iterate  $\mathbf{X}$ , it cannot exploit common sparsity in  $\mathbf{C}$  and the constraint matrices as well as dual methods or nonlinear-programming based methods. Thus our software may not be able to handle dense or sparse semidefinite blocks (with a single block) with order more than 3,000 on an average PC available in 2010.
- (c) Our interior-point algorithms are designed based on the existence of a central path in the interior of the primal-dual feasible region of  $(P)$  and  $(D)$ . For problems where the primal-dual feasible region is non-empty but has an empty interior, our SQLP solver can generally still deliver a reasonably good approximate optimal solution, but the solver tends to encounter numerical difficulties before a high accuracy solution can be obtained.

### 25.3.4 *The Structure Array OPTIONS for Parameters*

`sqlp.m` uses a number of parameters which are specified in a MATLAB structure array called `OPTIONS` in the m-file `sqlparameters.m`. If desired, the user can change the values of these parameters. The meaning of the specified fields in `OPTIONS` are given in the m-file itself. As an example, if the user does not wish to use corrector steps in Algorithm IPC, then he/she can do so by setting `OPTIONS.predcorr = 0`. If the user wants to use a fixed value, say 0.98, for the step-length parameter  $\bar{\gamma}$  in Algorithm IPC instead of the adaptive strategy used in the default, he/she can achieve that by setting `OPTIONS.gam = 0.98`. Similarly, if the user wants to solve the SQLP problem to an accuracy tolerance of  $1e-4$  instead of the default value of  $1e-8$  while using the default values for all other parameters, he/she only needs to set `OPTIONS.gaptol = 1e-4`.

The defaults in `sqlparameters.m` assume that the parameters  $v_j^s, v_i^q, v_k^l$  in  $(P)$  are all 0. For an SQLP problem where some of the parameters  $v_j^s, v_i^q, v_k^l$  are positive, the user needs to specify an  $L \times 1$  cell array `OPTIONS.parbarrier` to store the

parameters (including zeros) as follows. If the  $j$ th block is a semidefinite block consisting of one or more sub-blocks, say  $p$  of them, of orders  $s_{j1}, s_{j2}, \dots, s_{jp}$ , then

$$\text{OPTIONS.parbarrier}\{j\} = [v_{j1}^s, v_{j2}^s, \dots, v_{jp}^s].$$

If the  $i$ th block is a quadratic block consisting of one or more sub-blocks, say  $p$  of them, of dimensions  $q_{i1}, q_{i2}, \dots, q_{ip}$ , then  $\text{OPTIONS.parbarrier}\{i\} = [v_{i1}^q, \dots, v_{ip}^q]$ . If the  $k$ th block is the linear block, then  $\text{OPTIONS.parbarrier}\{k\} = [v_1^l, \dots, v_{n_l}^l]$ , while if the  $k$ th block is the unrestricted block, then  $\text{OPTIONS.parbarrier}\{k\} = \text{zeros}(1, n_u)$ . The reader is referred to Sect. 25.4.3 for an example where the objective function in (D) is given by  $\log \det(z^s)$ .

### 25.3.5 Running Problems in SDPA and SeDuMi Format

We provide two m-files, `read_sdpa.m` and `read_sedumi.m`, to respectively convert problem data stored in SDPA [37] and SeDuMi [26] format into MATLAB cell arrays described above. For an user who has the problem data generated in SDPT3 format, he/she can convert it to the SeDuMi format by using the function `SDPT3data_SEDUMIdata.m`.

The subdirectory `sdplib` in SDPT3 contains a few problems in SDPA format that are extracted from the SDPLIB library [2], while the subdirectory `dimacs` contains problems in SeDuMi format that are extracted from the DIMACS library [24]. Assuming that the current directory is SDPT3-4.0, we can read in and run the test problem `mcp250-1.dat-s` in the subdirectory `sdplib` as follows:

```
>> startup % set up Matlab paths
>> [blk,At,C,b] = read_sdpa('./sdplib/mcp250-1.dat-s');
>> [obj,X,y,Z,info] = sqlp(blk,At,C,b);
```

We can solve a DIMACS test problem in a similar manner.

```
>> OPTIONS.vers = 2; % use NT direction
>> [blk,At,C,b,perm] = read_sedumi('./dimacs/nb.mat');
>> [obj,X,y,Z,info] = sdpt3(blk,At,C,b,OPTIONS);
>> [x,y,z] = SDPT3soln.SEDUMIsoln(blk,X,y,Z,perm);
```

In the above, we call the hybrid solver `sdpt3.m` to solve the SQLP, and in the last line, we convert the solution in SDPT3 format to SeDuMi format.

### 25.3.6 Stopping Criteria

We define

$$n = \sum_{\{j: v_j^s = 0\}} s_j + \sum_{\{i: v_i^q = 0\}} q_i + |\{k : v_k^l = 0\}| \quad (25.3)$$

$$\mu(x, z) = \frac{1}{n} \sum_{\alpha \in \{s, q, l\}} \sum_{j=1}^{n_\alpha} \begin{cases} \langle x_j^\alpha, z_j^\alpha \rangle & \text{if } v_j^\alpha = 0 \\ 0 & \text{otherwise.} \end{cases} \quad (25.4)$$

$$\begin{aligned} \text{gap} &= \langle x, z \rangle - \sum_{\{j: v_j^s > 0\}} v_j^s (s_j + \log \det(x_j^s z_j^s / v_j^s)) \\ &\quad - \sum_{\{i: v_i^q > 0\}} v_i^q (1 + \log(\gamma(x_i^q) \gamma(z_i^q) / v_i^q)) - \sum_{\{k: v_k^l > 0\}} v_k^l (1 + \log(x_k^l z_k^l / v_k^l)). \end{aligned} \quad (25.5)$$

Note that if  $n = 0$ , we define  $\mu(x, z) = 0$ .

The algorithm is stopped when any of the following cases occur.

1. Solutions with the desired accuracy have been obtained, i.e.,

$$\phi := \max \{\text{relgap}, \text{pinfeas}, \text{dinfeas}\} \leq \text{OPTIONS.gaptol}, \quad (25.6)$$

where

$$\begin{aligned} \text{relgap} &= \frac{\text{gap}}{1 + |\langle c, x \rangle| + |b^T y|}, \quad \text{pinfeas} = \frac{\|\mathcal{A}(x) - b\|}{1 + \|b\|}, \\ \text{dinfeas} &= \frac{\|\mathcal{A}^T(y) + z - c\|}{1 + \|c\|}. \end{aligned}$$

2. Primal infeasibility is suggested because

$$b^T y / \|\mathcal{A}^T y + z\| > 10^8;$$

3. Dual infeasibility is suggested because

$$-c^T x / \|\mathcal{A}x\| > 10^8;$$

4. Slow progress is detected, measured by a rather complicated set of tests including

$$\text{relgap} < \max\{\text{pinfeas}, \text{dinfeas}\};$$

5. Numerical problems are encountered, such as the iterates not being positive definite or the Schur complement matrix not being positive definite; or
6. The step sizes fall below  $10^{-6}$ .

### 25.3.7 Initial Iterates

Our algorithms can start with an infeasible starting point. However, the performance of these algorithms is quite sensitive to the choice of the initial iterate. As observed in [11], it is desirable to choose an initial iterate that at least has the same order of magnitude as an optimal solution of the SLP. If a feasible starting point is not known, we recommend that the following initial iterate be used:  $y^0 = 0$ ,

$$(x_j^s)^0 = \xi_j^s I_{s_j}, (z_j^s)^0 = \eta_j^s I_{s_j}, (x_i^q)^0 = \xi_i^q e_i^q, (z_i^q)^0 = \eta_i^q e_i^q,$$

$$(x^l)^0 = \xi^l e^l, (z^l)^0 = \eta^l e^l, (x^u)^0 = 0,$$

where  $j = 1, \dots, n_s$ ,  $i = 1, \dots, n_q$ ,  $I_{s_j}$  is the identity matrix of order  $s_j$ ,  $e_i^q$  is the first  $q_i$ -dimensional unit vector,  $e^l$  is the  $l$ -vector of all ones, and

$$\xi_j^s = \max \left\{ 10, \sqrt{s_j}, s_j \max_{1 \leq k \leq m} \frac{1 + |b_k|}{1 + \|a_{j,k}^s\|_F} \right\},$$

$$\eta_j^s = \max \left\{ 10, \sqrt{s_j}, \max \{ \|c_j^s\|_F, \|a_{j,1}^s\|_F, \dots, \|a_{j,m}^s\|_F \} \right\},$$

$$\xi_i^q = \max \left\{ 10, \sqrt{q_i}, \sqrt{q_i} \max_{1 \leq k \leq m} \frac{1 + |b_k|}{1 + \|A_i^q(k,:)\|} \right\},$$

$$\eta_i^q = \max \left\{ 10, \sqrt{q_i}, \max \{ \|c_i^q\|, \|A_i^q(1,:)\|, \dots, \|A_i^q(m,:)\| \} \right\},$$

$$\xi^l = \max \left\{ 10, \sqrt{n_l}, \sqrt{n_l} \max_{1 \leq k \leq m} \frac{1 + |b_k|}{1 + \|A^l(k,:)\|} \right\},$$

$$\eta^l = \max \left\{ 10, \sqrt{n_l}, \max \{ \|c^l\|, \|A^l(1,:)\|, \dots, \|A^l(m,:)\| \} \right\}.$$

By multiplying the identity matrix  $I_{s_j}$  by the factors  $\xi_j^s$  and  $\eta_j^s$  for the semidefinite blocks, and similarly for the quadratic and linear blocks, the initial iterate has a better chance of having the appropriate order of magnitude. The initial iterate above is set by calling `infeaspt.m`, with syntax

```
[X0,y0,Z0] = infeaspt(blk,At,C,b,options,scalefac),
```

where `options = 1` (default) corresponds to the initial iterate just described, and `options = 2` corresponds to the choice where the blocks of  $\mathbf{X}\emptyset$ ,  $\mathbf{Z}\emptyset$  are `scalefac` times identity matrices or unit vectors, and  $\mathbf{y}\emptyset$  is a zero vector.

### 25.3.8 Preprocessing

#### 25.3.8.1 Nearly Dependent Constraints

The primal-dual path-following algorithm we implemented assumes that the matrix  $A$  in (25.2) has full column rank. But in our software, the presence of (nearly) dependent constraints is detected automatically, and warning messages are displayed if such constraints exist. When this happens, the user has the option of removing these (nearly) dependent constraints by calling a preprocessing routine to remove them by setting `OPTIONS.rmddepconstr = 1`. The routine (`checkdepconstr.m`) we have coded to detect nearly dependent constraints is based on computing the sparse  $LDL^T$  factorization of  $AA^T$ . Such a method is fast but is not as reliable as the method that is based on sparse  $LU$  factorization of  $A$ .

#### 25.3.8.2 Detecting Diagonal Blocks

We provide the m-file, `detect_lblk.m`, to look for diagonal blocks in semidefinite blocks in the data: see Sect. 25.2.1 for the use of this subroutine.

#### 25.3.8.3 Detecting Unrestricted Blocks

We have provided a routine, `detect_ublk.m`, to detect unrestricted variables that have been modelled as the difference of two nonnegative variables.

#### 25.3.8.4 Complex Data

In earlier versions, 2.3 or earlier, SDPT3 could directly handle complex data in SDP, i.e., the case where the constraint matrices are hermitian matrices. However, as problems with complex data rarely occur in practice, and in an effort to simplify the code, we removed this flexibility from subsequent versions.

Users can still solve an SDP with complex data using SDPT3-4.0. This is done by calling the m-file `convertcmplxsdp.m` to convert the SDP into one with real data. But unlike the earlier versions, here we convert the problem into one with real data

by doubling the size of the constraint matrices. Let  $B$  be an  $n \times n$  hermitian matrix. The conversion is based on the following equivalence:

$$B \text{ is positive semidefinite} \Leftrightarrow \begin{bmatrix} B^R & -B^I \\ B^I & B^R \end{bmatrix} \text{ is positive semidefinite,}$$

where  $B^R$  and  $B^I$  denote the real and imaginary parts of  $B$ , respectively. Note that since  $B$  is hermitian,  $B^R$  is symmetric and  $B^I$  is skew-symmetric.

Now suppose  $C, A_1, \dots, A_m$  are given  $n \times n$  hermitian matrices. Then  $C - \sum_{k=1}^m y_k A_k \geq 0$  if and only

$$\begin{bmatrix} C^R & -C^I \\ C^I & C^R \end{bmatrix} - \sum_{k=1}^m y_k^R \begin{bmatrix} A_k^R & -A_k^I \\ A_k^I & A_k^R \end{bmatrix} - \sum_{k=1}^m y_k^I \begin{bmatrix} -A_k^I & -A_k^R \\ A_k^R & -A_k^I \end{bmatrix} \geq 0. \quad (25.7)$$

Notice that the matrices  $[-A_k^I, -A_k^R; A_k^R, -A_k^I]$  are skew-symmetric. For a complex SDP, the vector  $b$  must necessarily be real, and the linear term in the objective function in (D) is replaced by  $\langle b, y^R \rangle$ . Since the skew symmetric matrices in (25.7) do not affect the positiveness condition and  $y^I$  does not appear in the objective function in (D), we can take  $y_k^I = 0, k = 1, \dots, m$ .

Note that the conversion of a complex SDP into a real SDP based on (25.7) would double the storage and if the data is dense, the cost of each interior-point iteration for solving the resulting real SDP is about twice as expensive as that for solving the complex SDP directly.

To convert an SDP with complex data into one with only real data, the m-file `convertcmpsdp.m` has the calling syntax:

```
[bblk, AAt, CC, bb] = convertcmpsdp(blk, At, C, b);
```

where `AAt` corresponds to the  $m$  real symmetric constraint matrices in the first summation in (25.7), `CC` corresponds to the real constant matrix in (25.7), and `bb` =  $b^R$ .

Internally, SDPT3 would automatically detect the presence of complex data in an SDP and convert it into one with only real data before solving it. But note that the current version of SDPT3 does not allow complex data corresponding to the quadratic (second-order cone) block.

### 25.3.8.5 Rotated Cones

Let  $K_r^n$  ( $n \geq 3$ ) be the rotated cone defined by

$$K_r^n = \{x^r = [u; v; w] \in \mathbf{R}^n : \|w\|^2 \leq 2uv, u, v \geq 0\}.$$

Note the constant “2” above. Define the symmetric orthogonal matrix  $T_n \in \mathbb{R}^{n \times n}$  as follows:

$$T_n = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & & \\ 1/\sqrt{2} & -1/\sqrt{2} & & \\ & & I_{n-2} & \end{bmatrix}.$$

It is easy to show that  $x^r \in K_r^n$  if and only if  $x^q := T_n x^r \in K_q^n$ , i.e.,  $T_n K_r^n = K_q^n$ . Thus we can always convert a rotated cone variable into one belonging to a second-order cone.

In SDPT3-4.0, the user can code a rotated cone block consisting of several sub-blocks, say  $p$  of them of dimension  $r_{i1}, \dots, r_{ip}$ , as follows:

```
blk{i,1}='r'; blk{i,2}=[r_{i1}, ..., r_{ip}];
```

Let  $D$  be the block diagonal matrix defined by  $D = \text{diag}(T_{r_{i1}}, \dots, T_{r_{ip}})$ . Internally, SDPT3 would convert such a rotated cone block and its associated data into a second-order cone block as follows:

```
blk{i,1}='q'; blk{i,2}=[r_{i1}, ..., r_{ip}];
```

```
At{i,1}=D*At{i,1}; C{i,1}=D*C{i,1};
```

## 25.4 Examples

For an user to solve his SQLP problem using SDPT3, the first task he/she needs to perform is to write his problem in the standard form given in  $(P)$  and then to code the problem data corresponding to the standard form. In the last few years, modelling languages such as CVX [12] and YALMIP [19] have been developed for advanced modelling and solution of convex optimization problems. These modelling languages would automatically convert an SQLP problem which is not necessarily expressed in the standard form given in  $(P)$  or  $(D)$  into the required standard form for solvers such as SDPT3 or SeDuMi. They would also generate the problem data corresponding to the standard form and call a solver to solve the problem. In CVX, SDPT3 is used as the default solver, but SeDuMi is also conveniently available. Similarly, YALMIP can also use SDPT3 or SeDuMi as its computational engine to solve SQLP problems.

For users who are interested only in solving small SQLP problems, we would advise them to use CVX or YALMIP to convert their problems into the required standard form and to generate the corresponding data as the process is usually quite tedious though mechanical. But for users who wish to solve large SQLP problems, it is sometimes highly beneficial for the users themselves to convert their

problems into the required standard form rather than relying on CVX or YALMIP as these modelling languages may introduce large number of additional variables or significantly expand the problem dimensions when converting the problems into the standard form since they are based on automatic conversion procedures which do not take into account problem structures that users may be able to exploit.

The simplest way to learn how to convert an SQLP problem into the standard form given in  $(P)$  or  $(D)$  and to generate the corresponding problem data in SDPT3 format is through examples. (Note that the user can also store the problem data in either the SDPA or SeDuMi format, and then use the m-files `read_sdpa.m` or `read_sedumi.m` to read the data into SDPT3.) The subdirectory `Examples` in SDPT3 contains many such example files, including those for the nearest correlation matrix problem and the complex data norm minimization problem. Here we will just mention a few.

### 25.4.1 Conversion of Problems into the Standard Form

As mentioned before, SQLP problems are usually not formulated in the standard form  $(P)$  or  $(D)$ , and it is often quite tedious to convert such problems into the standard form. As such, users who are solving small to medium scale SQLPs are encouraged to use modelling languages such as CVX [12] or YALMIP [19] to automate the conversion process. To give the user an idea on how the conversion is done, here we shall just give an example to show how an SDP with linear equality and inequality constraints can be converted into the standard form given in  $(P)$ . Suppose we have an SDP of the following form:

$$(P_1) \quad \min\{\langle c^s, x^s \rangle : \mathcal{A}^s(x^s) = b, \quad \mathcal{B}^s(x^s) \leq d, \quad x^s \in K_s^n\}.$$

where  $d \in R^p$  and  $\mathcal{B}^s$  is a linear map. By introducing a slack variable  $x^l$ , we can easily convert  $(P_1)$  into standard form, namely,

$$\begin{aligned} (P_1^*) \quad & \min \left\{ \langle c^s, x^s \rangle + \langle c^l, x^l \rangle : \mathcal{A}^s(x^s) \right. \\ & \left. = b, \quad \mathcal{B}^s(x^s) + B^l x^l = d, \quad x^s \in K_s^n, x^l \in K_l^p \right\}, \end{aligned}$$

where  $c^l = 0$ , and  $B^l = I_{p \times p}$ . With our use of cell arrays to store SQLP data, it is easy to take the problem data of  $(P_1)$  and use them for the standard form  $(P_1^*)$  as follows:

```
blk{1,1}='s'; blk{1,2}=n; At{1,1}=[A^s; B^s]^T; C{1,1}=c^s;
blk{2,1}='l'; blk{2,2}=p; At{2,1}=[sparse(m,p); speye(p,p)]^T;
C{2,1}=c^l;b = [b; d];
```

In the subdirectory, an SDP of the form  $(P_1)$  is coded in the example file `max_kcut.m`.

### 25.4.2 The MAXCUT Problem

Let  $\mathcal{S}_+^n$  be the space of  $n \times n$  symmetric positive semidefinite matrices. Let  $B$  be the weighted adjacency matrix of a graph. The SDP relaxation of the MAXCUT problem associated with  $B$  has the following form:

$$\min \{ \langle C, X \rangle : \text{diag}(X) = e, \quad X \in \mathcal{S}_+^n \},$$

where  $e$  is the vector of ones, and  $C = -(\text{Diag}(Be) - B)/4$ . It is clear that we need the cell array,  $\text{blk}\{1,1\} = 's'$ ,  $\text{blk}\{1,2\} = n$ , to record the block structure of the problem. The constraint matrices can be constructed conveniently via an  $1 \times n$  cell array as follows:

```
for k=1:n; AA{1,k}=spconvert([k,k,1;n,n,0]); end;
At = svec(blk,AA);
```

For more details, see the m-file `maxcut.m` in the subdirectory `Examples`. (We could also create a version of the problem explicitly showing the low-rank structure; however, as the constraint matrices are so sparse, this would not be more efficient.)

### 25.4.3 D-Optimal Experiment Design: An Example with an Explicit Barrier Term

Given a set of points  $\{v_1, \dots, v_p\}$  in  $R^n$  with  $n \leq p$ , the D-optimal experiment design problem [30, 35] needs to solve the following dual SQLP:

$$\begin{aligned} & \max \log \det(Z) \\ \text{s.t. } & \sum_{k=1}^p y_k (-v_k v_k^T) + Z = 0, \quad Z \in \mathcal{S}_{++}^n \\ & -y + z^l = 0, \quad z^l \in R_+^p \\ & e^T y = 1, \quad y \in R^p. \end{aligned}$$

The associated problem data can be coded in SDPT3 format as follows:

```
b = zeros(p,1);
blk{1,1} = 's'; blk{1,2} = n;
AA = cell(1,p); for k=1:p; AA{k} = -vk*vk'; end
At(1) = svec(blk(1,:),AA); C{1,1} = sparse(n,n);
blk{2,1} = 'l'; blk{2,2} = p; At{2,1} = -speye(p);
C{2,1}=zeros(p,1);
blk{3,1} = 'u'; blk{3,2} = 1; At{3,1} = ones(1,p);
C{3,1}=1;
```

Because the problem contains an explicit log-barrier term in the objective function, we also need to set up `OPTIONS.parbarrier` as follows:

```
OPTIONS.parbarrier{1,1}=1; OPTIONS.parbarrier{2,1}=0;
OPTIONS.parbarrier{3,1}=0;
```

For more details, see the m-file `Doptdesign.m` in the subdirectory `Examples`.

The constraint matrices corresponding to the semidefinite block in this example are all rank-one matrices. The user can explicitly code such structures for SDPT3 as follows:

```
blk{1,1} = 's'; blk{1,2} = n; blk{1,3} = ones(1,p);
At{1,1} = []; At{1,2} = [v1,...,vp]; At{1,3} = -ones(p,1);
```

As an example on the potential speed up one may gain by exploiting the low-rank structure in the constraint matrices, we compare the times taken to solve two D-optimal experiment design problems with randomly generated data  $\{v_1, \dots, v_p\}$  in  $R^n$ . For  $n = 200$ ,  $p = 400$  ( $n = 100$ ,  $p = 400$ ), the time taken to solve the problem without exploiting the low-rank structure is 157.1 (44.2) seconds, compared to just 2.9 (2.1) seconds when the low-rank structure is exploited.

#### 25.4.4 An LMI Example

Consider the following LMI problem [3]:

$$\max_{\eta \in \mathbb{R}, Y \in \mathcal{S}^n} \left\{ -\eta : GY + YG^T \leq 0, -Y \leq -I, Y - \eta I \leq 0, Y_{11} = 1 \right\} \quad (25.8)$$

where  $G \in R^{n \times n}$ . This problem can be viewed as a dual SDP with  $Y$  identified as a vector  $y$  in  $R^{n(n+1)/2}$ . In this case, we have  $(A_1^s)^T y = \text{svec}(G\text{smat}(y) + \text{smat}(y)G^T)$ , where `smat` is the inverse of `svec`. The SDP data can be generated for SDPT3 as follows:

```
blk{1,1} = 's'; blk{1,2} = n; blk{2,1} = 's'; blk{2,2} = n;
blk{3,1} = 's'; blk{3,2} = n; blk{4,1} = 'u'; blk{4,2} = 1;
n2 = n*(n+1)/2; zz = sparse(n2,1); I = speye(n);
At{1,1} = [lmifun(G,I), zz]; At{2,1} = [-lmifun(I/2,I), zz];
At{3,1} = [lmifun(I/2,I), svec(blk(1,:))-I];
At{4,1} = [1, zz']; C{1,1} = sparse(n,n); C{2,1} = -I;
C{3,1} = sparse(n,n); C{4,1} = 1; b = [zz; -1];
```

In the above, `lmi fun(G, H)` is a function (available in `Examples`) that generates the matrix representation of the linear map  $y \in \mathbf{R}^{n(n+1)/2} \mapsto \text{svec}(G\text{smat}(y)H^T + H\text{smat}(y)G^T)$ .

For more details, see the m-file `lmi examp1.m` in the subdirectory `Examples`.

### 25.4.5 A Problem with a Convex Quadratic Objective Function

The nearest correlation matrix problem (see `corrmat.m` in the `Examples` subdirectory) has a convex quadratic objective function in the primal problem. Here we consider the case where the dual problem has a concave quadratic objective function, namely,

$$\max_{y, Z} \left\{ b^T y - y^T H y : \mathcal{A}_s^T y + z_s = c_s, z_s \in \mathcal{S}_+^n, y \in \mathbf{R}^m \right\},$$

where  $H$  is a given symmetric positive semidefinite matrix. Suppose  $H = R^T R$  is the Cholesky factorization. Then the above problem can be rewritten as follows:

$$\max_{y, Z} \left\{ b^T y - t : \|Ry\|^2 \leq t, \mathcal{A}_s^T y + z_s = c_s, z_s \in \mathcal{S}_+^n, y \in \mathbf{R}^m \right\}. \quad (25.9)$$

It can in turn be expressed as a standard SQLP in dual form as follows:

$$\begin{aligned} & \max \quad [-1; b]^T [t; y] \\ \text{s.t.} \quad & \begin{bmatrix} -1/2 & 0 \\ -1/2 & 0 \\ 0 & R \end{bmatrix} [t; y] + z_q = \begin{bmatrix} 1/2 \\ -1/2 \\ 0 \end{bmatrix} \\ & [0, \mathcal{A}_s^T][t; y] + z_s = c_s, \quad z_q \in K_q^{m+2}, z_s \in \mathcal{S}_+^n, [t; y] \in \mathbf{R}^{m+1}. \end{aligned}$$

### 25.4.6 An Example from Distance Geometry

Consider a graph  $G = (V, \mathcal{E}, D)$  where  $V = \{1, 2, \dots, n\}$ ,  $\mathcal{E}$ , and  $D = (d_{ij})$  denote the nodes, edges, and associated weight matrix on the edges, respectively. The problem is to find points  $x_1, \dots, x_n$  in  $\mathbf{R}^p$  (for some  $p$ ) such that the pairwise Euclidean distance between  $x_i$  and  $x_j$  is as close as possible to  $d_{ij}$  if  $(i, j) \in \mathcal{E}$ . The associated minimization problem is the following:

$$\min \left\{ \sum_{(i,j) \in \mathcal{E}} \left| \|x_i - x_j\|^2 - d_{ij}^2 \right| : X := [x_1, \dots, x_n] \in \mathbf{R}^{p \times n} \right\}.$$

By noting that  $\|x_i - x_j\|^2 = e_{ij}^T X^T X e_{ij}$ , where  $e_{ij} = e_i - e_j$ , the above problem can equivalently be written as  $\min \left\{ \sum_{(i,j) \in \mathcal{E}} |\langle e_{ij} e_{ij}^T, Y \rangle - d_{ij}^2| : Y = X^T X, X \in \mathbb{R}^{p \times n} \right\}$ . One of the SDP relaxations of the above problem is  $\min \left\{ \sum_{(i,j) \in \mathcal{E}} |\langle e_{ij} e_{ij}^T, Y \rangle - d_{ij}^2| : Y \in \mathcal{S}_+^n \right\}$ , where the corresponding problem in standard form is given by

$$\min \left\{ \sum_{(i,j) \in \mathcal{E}} \alpha_{ij}^+ + \alpha_{ij}^- : \langle e_{ij} e_{ij}^T, Y \rangle - \alpha_{ij}^+ + \alpha_{ij}^- = d_{ij}^2, \alpha_{ij}^+, \alpha_{ij}^- \geq 0 \forall (i,j) \in \mathcal{E}, Y \in \mathcal{S}_+^n \right\}.$$

Let  $m = |\mathcal{E}|$ . The SQLP data can be coded as follows:

```

blk{1,1} = 's'; blk{1,2} = n;
AA = cell(1,m); b = zeros(m,1); cnt = 0;
for i = 1:n-1
    for j = i+1:n
        if (D(i,j) ~= 0)
            cnt = cnt + 1;
            AA{cnt} = spconvert([i,i,1; i,j,-1; j,i,-1; j,j,1;
                                  n,n,0]);
            b(cnt) = D(i,j)^2;
        end
    end
end
At{1} = svec(blk(1,:),AA); C{1,1} = sparse(n,n);
blk{2,1} = 'l'; blk{2,2} = 2*m;
At{2,1} = [-speye(m), speye(m)]; C{2,1} = ones(2*m,1);

```

## 25.5 Implementation Details

The main step at each iteration of our algorithms is the computation of the search direction  $(\Delta x, \Delta y, \Delta z)$  from the *symmetrized Newton equation* with respect to some invertible block diagonal scaling matrix that is usually chosen as a function of the current iterate  $x, z$ .

### 25.5.1 The HKM Search Direction

Let

$$J_i^q = \begin{bmatrix} 1, 0; 0, -I_{q_{i-1}} \end{bmatrix}, \quad (25.10)$$

For the choice of the HKM scaling matrix [14, 17, 21, 22, 34], the search direction  $(\Delta x, \Delta y, \Delta z)$  is obtained from the following system of equations:

$$\begin{aligned}
& \mathcal{A}^s(\Delta x^s) + A^q(\Delta x^q) + A^l(\Delta x^l) + A^u(\Delta x^u) \\
&= R_p := b - \mathcal{A}^s(x^s) - A^q(x^q) - A^l(x^l) - A^u(x^u) \\
& (\mathcal{A}^s)^T \Delta y + \Delta z^s = R_d^s, \quad (A^q)^T \Delta y + \Delta z^q = R_d^q, \\
& (A^l)^T \Delta y + \Delta z^l = R_d^l, \quad (A^u)^T \Delta y = R_d^u := c^u - (A^u)^T y \\
& \Delta x^s + H^s(\Delta z^s) = R_c^s := (\max\{\sigma\mu(x, z), \nu_j^s\}(z_j^s)^{-1} - x_j^s)_{j=1}^{n_s} \\
& \Delta x^q + H^q(\Delta z^q) = R_c^q := (\max\{\sigma\mu(x, z), \nu_i^q\}(z_i^q)^{-1} - x_i^q)_{i=1}^{n_q} \\
& \Delta x^l + H^l(\Delta z^l) = R_c^l := (\max\{\sigma\mu(x, z), \nu_k^l\}(z_k^l)^{-1} - x_k^l)_{k=1}^{n_l}
\end{aligned} \tag{25.11}$$

where  $\sigma \in (0, 1)$  is the centering parameter;  $R_d^\alpha = c^\alpha - z^\alpha - (\mathcal{A}^\alpha)^T y, \alpha \in \{s, q, l\}$ ;  $(z_j^s)^{-1}$  and  $(z_k^l)^{-1}$  have the usual meaning, and  $(z_i^q)^{-1} := J_i^q z_i^q / \gamma(z_i^q)^2$ . In the above,

$$\begin{aligned}
H^s(\Delta z^s) &= \left( H_j^s(\Delta z_j^s) := \frac{1}{2}((z_j^s)^{-1}) \Delta z_j^s x_j^s + x_j^s \Delta z_j^s (z_j^s)^{-1} \right)_{j=1}^{n_s} \\
H^q(\Delta z^q) &= (H_i^q(\Delta z_i^q) := -\frac{\langle x_i^q, z_i^q \rangle}{\gamma(z_i^q)^2} J_i^q \Delta z_i^q + x_i^q ((z_i^q)^{-1})^T \Delta z_i^q \\
&\quad + (z_i^q)^{-1} (x_i^q)^T \Delta z_i^q)_{i=1}^{n_q} \\
H^l(\Delta z^l) &= \text{Diag}(x^l) \text{Diag}(z^l)^{-1} \Delta z^l.
\end{aligned} \tag{25.12}$$

The search direction can be computed via a Schur complement equation as follows (the reader is referred to [1] and [27] for details). First compute  $\Delta y$  from the Schur complement equation

$$\underbrace{\begin{bmatrix} M & A^u \\ (A^u)^T & 0 \end{bmatrix}}_{\mathcal{M}} \begin{bmatrix} \Delta y \\ \Delta x^u \end{bmatrix} = \begin{bmatrix} h \\ R_d^u \end{bmatrix} \tag{25.13}$$

where

$$M = \sum_{j=1}^{n_s} \underbrace{\mathcal{A}_j^s H_j^s (\mathcal{A}_j^s)^T}_{M_j^s} + \sum_{i=1}^{n_q} \underbrace{A_i^q H_i^q (A_i^q)^T}_{M_i^q} + \underbrace{A^l H^l (A^l)^T}_{M^l} \tag{25.14}$$

$$h = R_p + \mathcal{A}^s(H^s(R_d^s) - R_c^s) + A^q(H^q(R_d^q) - R_c^q) + A^l(H^l(R_d^l) - R_c^l). \tag{25.15}$$

(The notation in (25.14) should be interpreted as follows: the  $k$ th columns of  $M_j^s$  and  $M_i^q$  are  $\mathcal{A}_j^s H_j^s(a_{j,k}^s)$  and  $A_i^q H_i^q(a_{i,k}^q)$ , with  $a_{i,k}^q$  the  $k$ th column of  $(A_{i,k}^q)^T$ . Note that the matrices  $M_j^s$ ,  $M_i^q$ ,  $M^l$  are all symmetric positive definite.) Then compute  $\Delta x$  and  $\Delta z$  from the equations

$$\begin{aligned}\Delta z^s &= R_d^s - (\mathcal{A}^s)^T \Delta y, & \Delta z^q &= R_d^q - (A^q)^T \Delta y, & \Delta z^l &= R_d^l - (A^l)^T \Delta y \\ \Delta x^s &= R_c^s - H^s(\Delta z^s), & \Delta x^q &= R_c^q - H^q(\Delta z^q), & \Delta x^l &= R_c^l - H^l(\Delta z^l).\end{aligned}$$

### 25.5.2 The NT Search Direction

The user also has the choice of using the NT direction [22, 23, 34]. Let  $w_j^s$  be the unique positive definite matrix such that  $w_j^s z_j^s w_j^s = x_j^s$ . Let

$$\omega_i^q = \sqrt{\frac{\gamma(z_i^q)}{\gamma(x_i^q)}}, \quad \xi_i^q = \frac{1}{\omega_i^q} z_i^q + \omega_i^q J_i^q x_i^q, \quad t_i^q = \frac{\sqrt{2}}{\omega_i^q \gamma(\xi_i^q)} J_i^q \xi_i^q. \quad (25.16)$$

In this case, the search direction  $(\Delta x, \Delta y, \Delta z)$  satisfies the same system as in (25.11) except that  $H^s$  and  $H^q$  are replaced by

$$\begin{aligned}H^s(\Delta z^s) &= (H_j^s(\Delta z_j^s) := w_j^s \Delta z_j^s w_j^s)_{j=1}^{n_s}, \\ H^q(\Delta z^q) &= \left( H_i^q(\Delta z_i^q) := -\frac{1}{(\omega_i^q)^2} J_i^q \Delta z_i^q + t_i^q (t_i^q)^T \Delta z_i^q \right)_{i=1}^{n_q}. \quad (25.17)\end{aligned}$$

### 25.5.3 Choice of Search Direction

The current version of the code allows only two search directions, HKM and NT. In older versions, version 2.3 or earlier, we also implemented the AHO direction of Alizadeh et al. [1] and the GT direction [29], where both directions tend to give more accurate results, but these are uncompetitive when the problems are of large scale.

For the HKM and NT search directions, our computational experience on problems tested in Sect. 25.8 is that the HKM direction is almost universally faster than NT on problems with semidefinite blocks, especially for sparse problems with large semidefinite blocks. The reason that the latter is slower is because computing the NT scaling matrix  $w_j^s$  requires a full eigenvalue decomposition. This computation can dominate the work at each interior-point iteration when the problem is sparse.

The NT direction, however, was faster on sparse SOCP problems. The reason for this behavior is not hard to understand. By comparing the formulae for  $H_i^q$  for the HKM and NT directions in (25.12) and (25.17), it is clear that more computation is required to assemble the Schur complement matrix and more low-rank updating is necessary for the former direction.

### 25.5.4 Computation of the Schur Complement Matrix

Generally, the most expensive part in each iteration of Algorithm IPC lies in the computation and factorization of the Schur complement matrix  $M$  defined in (25.13). And this depends critically on the size and density of  $M$ . Note that the density of this matrix depends on two factors: (1) The density of  $\mathcal{A}^s$ ,  $A^q$ , and  $A^l$ , and (2) any additional fill-in introduced because of the terms  $H^s$ ,  $H^q$ , and  $H^l$  in (25.14).

#### 25.5.4.1 Semidefinite Blocks

For problems with semidefinite blocks, the contribution by the  $j$ th semidefinite block to  $M$  is given by  $M_j^s := \mathcal{A}_j^s H_j^s (\mathcal{A}_j^s)^T$ . The matrix  $M_j^s$  is generally dense even if  $A_j^s$  is sparse. The computation of each entry of  $M_j^s$  involves matrix products, which has the form

$$(M_j^s)_{\alpha\beta} = \begin{cases} \langle a_{j,\alpha}^s, x_j^s a_{j\beta}^s (z_j^s)^{-1} \rangle & \text{for the HKM direction.} \\ \langle a_{j,\alpha}^s, w_j^s a_{j\beta}^s w_j^s \rangle & \text{for the NT direction.} \end{cases}$$

This computation can be very expensive if it is done naively without properly exploiting the sparsity that is generally present in the constraint matrices in  $\mathcal{A}_j^s$ . In our earlier papers [27, 31], we discussed briefly how sparsity of  $\mathcal{A}_j^s$  is exploited in our implementation by following the ideas presented by Fujisawa, Kojima, and Nakata in [11]. Further details on the efficient implementation of these ideas are given in [33].

When the constraint matrices have low-rank structures as described in Sect. 25.2.2, we can also compute the element  $(M_j^s)_{\alpha\beta}$  as follows:

$$(M_j^s)_{\alpha\beta} = \text{Tr}(\tilde{V}_\beta^T V_\alpha D_\alpha V_\alpha^T \widehat{V}_\beta D_\beta),$$

where  $\tilde{V}_\beta = x_j^s V_\beta$ , and  $\widehat{V}_\beta = (z_j^s)^{-1} V_\beta$  if the HKM direction is used; and  $\tilde{V}_\beta = \widehat{V}_\beta = w_j^s V_\beta$  if the NT direction is used. Assume for simplicity that all the constraint matrices associated with the  $j$ th semidefinite block are dense and low-rank, i.e.,  $p = 0$  in Sect. 25.2.2. Suppose that the matrices  $\tilde{V}_k$ ,  $\widehat{V}_k$ ,  $k = 1, \dots, m$ , are pre-computed (at the cost of  $\Theta(s_j^2 \sum_{k=1}^m r_{j,k})$  flops). Then it would take an additional  $\Theta(s_j(\sum_{k=1}^m r_{j,k})^2)$  flops to compute  $M_j^s$  since each element  $(M_j^s)_{\alpha\beta}$  can be computed

at  $\Theta(s_j r_{j,\alpha} r_{j,\beta})$  flops. In contrast, without exploiting the low-rank structures, it would take  $\Theta(s_j^3 m) + \Theta(s_j^2 m^2)$  flops to compute  $M_j^s$ . If the average rank of the constraint matrices is  $r$ , then the latter complexity is  $\Theta(s_j/r^2)$  times larger than the former of  $\Theta(s_j^2 m r) + \Theta(s_j m^2 r^2)$ . Thus it is definitely advantageous to exploit low-rank structures.

As an example, we generated a random SDP with low rank structure using the m-file `randlowranksdp.m` described in Sect. 25.2.2 with  $n = 200$  and  $m = 1,000$ ; the solver `sqlp.m` ran about 6 times faster when the low-rank structure was exploited.

#### 25.5.4.2 Quadratic and Linear Blocks

For the linear block,  $H^l$  is a diagonal matrix and it does not introduce any additional fill-in. This matrix does, however, affect the conditioning of the Schur complement matrix.

From (25.14), it is easily shown that the contribution of the quadratic blocks to the matrix  $M$  is given by

$$M_i^q = \begin{cases} -\frac{\langle x_i^q, z_i^q \rangle}{\gamma^2(z_i^q)} A_i^q J_i^q (A_i^q)^T + u_i^q (v_i^q)^T + v_i^q (u_i^q)^T, & \text{for HKM direction} \\ -\frac{1}{(\omega_i^q)^2} A_i^q J_i^q (A_i^q)^T + w_i^q (w_i^q)^T & \text{for NT direction} \end{cases} \quad (25.18)$$

where  $u_i^q = A_i^q x_i^q$ ,  $v_i^q = A_i^q (z_i^q)^{-1}$ ,  $w_i^q = A_i^q t_i^q$ .

In the rest of this subsection, we focus our discussion on the HKM direction, but the same holds true for the NT direction.

The appearance of the outer-product terms in the equation above is potentially alarming. If the vectors  $u_i^q$ ,  $v_i^q$  are dense, then even if  $A_i^q$  is sparse, the corresponding matrix  $M_i^q$ , and hence the Schur complement matrix  $M$ , will be dense. A direct factorization of the resulting dense matrix will be very expensive for even moderately large  $m$ .

The density of the matrix  $M_i^q$  depends largely on the particular problem structure. When the problem has many small quadratic blocks, it is often the case that each block appears in only a small fraction of the constraints. In this case, all  $A_i^q$  matrices are sparse and the vectors  $u_i^q$  and  $v_i^q$  turn out to be sparse vectors for each  $i$ . Consequently, the matrices  $M_i^q$  remain relatively sparse for these problems. As a result,  $M$  is also sparse and it can be factorized directly with reasonable cost. This behavior is typical for all `nql` and `qssp` problems from the DIMACS library.

The situation is drastically different for problems where one of the quadratic blocks, say the  $i$ th block, is large. For such problems the vectors  $u_i^q$ ,  $v_i^q$  are typically dense, and therefore,  $M_i^q$  is likely be a dense matrix even if the data  $A_i^q$  is sparse. However, observe that  $M_i^q$  is a rank-two perturbation of a sparse matrix when  $A_i^q (A_i^q)^T$  is sparse. In such a situation, it is advantageous to use the dense-column handling technique described in Sect. 25.5.7 to reduce the computational cost in

solving (25.13). This approach helps tremendously on the scheduling problems from the DIMACS library.

To apply the dense-column handling technique, we need to modify the sparse portion of the matrix  $M_i^q$  slightly. Since the diagonal matrix  $-J_i$  has a negative component, the matrix  $-A_i^q J_i^q (A_i^q)^T$  need not be a positive definite matrix, and therefore the Cholesky factorization of the sparse portion of  $M_i^q$  need not exist. To overcome this difficulty, we use the following identity:

$$M_i^q = \frac{\langle x_i^q, z_i^q \rangle}{\gamma^2(z_i^q)} A_i^q (A_i^q)^T + u_i^q (v_i^q)^T + v_i^q (u_i^q)^T - k_i k_i^T, \quad (25.19)$$

where  $k_i = \sqrt{2} \langle x_i^q, z_i^q \rangle / \gamma^2(z_i^q) A_i^q(:, 1)$ . Note that if  $A_i^q$  is a large sparse matrix with a few dense columns, we can also explicitly separate the outer-product terms contributed by these dense columns from the sparse part of  $A_i^q (A_i^q)^T$  in (25.19).

### 25.5.5 Solving the Schur Complement Equation

The linear system (25.13) typically becomes more and more ill-conditioned as  $\mu(x, z)$  decreases to 0. Thus iterative refinement is generally recommended to improve the accuracy of the computed solution. An even better approach to solve (25.13) is via a preconditioned symmetric quasi-minimal residual method (PSQMR) [10] with the preconditioner computed based on the following analytical expression of  $\mathcal{M}^{-1}$ :

$$\mathcal{M}^{-1} = \begin{bmatrix} M^{-1} - M^{-1} A^u S^{-1} (A^u)^T M^{-1} & M^{-1} A^u S^{-1} \\ S^{-1} (A^u)^T M^{-1} & -S^{-1} \end{bmatrix}, \quad (25.20)$$

where  $S = (A^u)^T M^{-1} A^u$ . Note that for a given vector  $[u; v]$ ,  $\mathcal{M}^{-1}[u; v]$  can be evaluated efficiently as follows:

$$\hat{u} = M^{-1} u, \quad t = S^{-1} ((A^u)^T \hat{u} - v), \quad \mathcal{M}^{-1}[u; v] = [\hat{u} - M^{-1} A^u t; t].$$

Thus if the Cholesky factorization of  $M$  and that of  $S$  are computed, then each evaluation involves solving four triangular linear systems for  $M$  and two triangular linear systems for  $S$ .

We should mention that state-of-the-art Cholesky factorization software is highly developed and optimized. Thus our preference is to solve a linear system via Cholesky factorizations whenever possible. For most SDP problems, the matrix  $M$  is typically dense even when the constraint matrices are sparse. In this case, we use the routine `chol` (based on the LAPACK routine `dpotrf`) in MATLAB to compute the Cholesky factorization of a dense matrix.

For most sparse SOCP problems, the matrix  $M$  is usually sparse after dense-column handling. Let  $M_{sp}$  be the sparse part of  $M$  after dense-column handling. In this case, the Cholesky factorization routine `chol` for a dense matrix is not efficient enough since it does not exploit sparsity. To factorize the sparse matrix  $M_{sp}$  more efficiently, we use the sparse cholesky solver `cholmod` of Davis [5], which is available in MATLAB as `chol`. In earlier versions, we used a C translation of the Fortran programs developed by Ng, Peyton, and Liu for sparse Cholesky factorization [18].

The effect of using a sparse Cholesky solver for sparse SOCP problems was dramatic. We observed speed-ups of up to two orders of magnitude. In our implementation, SDPT3 automatically makes a choice between MATLAB's built-in `chol` routine and the sparse Cholesky solver based on the density of  $M$ . The cutoff density is specified in the parameter `OPTIONS.spdensity`.

The approach of solving (25.13) by the SQMR method with preconditioner (25.20) works reasonably well if the following conditions are satisfied: (1) the number of columns of  $A^u$  is small and  $A^u$  is well-conditioned; (2) the matrix  $M$  is not extremely ill-conditioned. (3) the matrix  $S$  is not extremely ill-conditioned. However, when these conditions are not satisfied, preconditioning (25.13) based on (25.20) may not be advisable because either (a) computing  $S$  becomes very expensive due to large number of columns in  $A^u$ , or (b) the computed preconditioner based on (25.20) is no longer an accurate approximation of  $\mathcal{M}^{-1}$ . Note that  $S$  is typically much more ill-conditioned than  $A^u$ , especially when  $A^u$  is ill-conditioned. When conditions (1)–(3) are not satisfied, it is advisable to directly use an  $LU$  or  $LDL^T$  factorization of the symmetric indefinite matrix  $\mathcal{M}$  to compute an approximation of  $\mathcal{M}^{-1}$ . As the matrix  $\mathcal{M}$  is usually highly ill-conditioned, we observed that the computed solution based on  $LU$  factorization is typically more accurate than the one computed based on  $LDL^T$  factorization. Thus, even though  $LU$  is twice as expensive as  $LDL^T$  factorization when the matrix is dense, we simply use the MATLAB routine `lu` to compute an  $LU$  factorization of  $\mathcal{M}$ , and use the computed  $LU$  factors to precondition the SQMR iterative method used to solve (25.13).

Again, in our implementation, SDPT3 automatically makes a choice on whether to compute a dense or sparse  $LU$  factorization based on the density of  $\mathcal{M}$ . In the case of sparse  $LU$  factorization, SDPT3 uses the UMFPACK package of Davis [6], which is available in MATLAB under the `lu` command. We should mention that we have tested the sparse  $LDL^T$  factorization of the symmetric matrix  $\mathcal{M}$  (when it is a sparse) based on the MA57 routine [7] (available in MATLAB under the `ldl` command) of the Harwell subroutine library [13]. But we have found that it is not as efficient as the sparse  $LU$  factorization based on the UMFPACK package when the matrix is highly ill-conditioned. More importantly, the computed solution based on sparse  $LDL^T$  factorization is often not as accurate as the one computed based on sparse  $LU$  factorization.

### 25.5.6 Internal Handling of Unrestricted Blocks

As mentioned in the last subsection, solving the symmetric indefinite system (25.13) can potentially be very expensive when  $A^u$  is ill-conditioned or has a large number of columns because computing the sparse LU factorization of a sparse matrix can be much more costly than that for a symmetric positive definite matrix of the same order. It is possible to avoid the need to solve a symmetric indefinite system if we reformulate the equality constraint in ( $D$ ) as

$$(A^u)^T y + z_+^u = c^u, \quad z_+^u \geq 0, \quad -(A^u)^T y + z_-^u = -c^u, \quad z_-^u \geq 0,$$

with the corresponding primal variable  $x^u$  expressed as

$$x^u = x_+^u - x_-^u, \quad x_+^u, x_-^u \geq 0.$$

In this case, the system (25.13) is replaced by

$$\left( M + A^u \text{Diag}(x_+^u) \text{Diag}(z_+^u)^{-1} (A^u)^T + A^u \text{Diag}(x_-^u) \text{Diag}(z_-^u)^{-1} (A^u)^T \right) \Delta y = \text{rhs}$$

where rhs denotes the right-hand-side vector. Notice that in contrast to (25.13), the coefficient matrix is now symmetric positive definite.

But such a reformulation is not without difficulties. In fact, the variables  $x_+^u, x_-^u$  tend to become very large and  $z_+^u, z_-^u$  tend to become extremely small as the interior-point iteration progresses, and this generally makes the component matrices,  $A^u \text{Diag}(x_+^u) \text{Diag}(z_+^u)^{-1} (A^u)^T$  and  $A^u \text{Diag}(x_-^u) \text{Diag}(z_-^u)^{-1} (A^u)^T$ , extremely ill-conditioned. Fortunately, the following heuristic to modify the vectors  $x_+^u, x_-^u$  can typically ameliorate such an ill-conditioning problem:

$$x_+^u := x_+^u - 0.8 \min(x_+^u, x_-^u), \quad x_-^u := x_-^u - 0.8 \min(x_+^u, x_-^u).$$

This modification does not change the original variable  $x^u$  but does slow down the growth of  $x_+^u, x_-^u$ . After these modified vectors have been obtained, we also add positive perturbations to the vectors  $z_+^u, z_-^u$ . Such a modification in  $z_+^u, z_-^u$  ensures that they approach 0 at the same rate as  $\mu$ , and thus prevents the dual problem ( $D$ ) from approaching the equality constraint too closely prematurely.

For computational efficiency, in the current implementation of SDPT3, we always reformulate an unrestricted vector by the difference of two non-negative vectors.

### 25.5.7 Dense-Column Handling

Here we describe our technique to handle dense columns when  $M$  is a low-rank perturbation of a sparse matrix. In such a case, the Schur complement matrix  $M$  can

be written in the form

$$M = M_{\text{sp}} + UDU^T \quad (25.21)$$

where  $M_{\text{sp}}$  is a sparse symmetric positive semidefinite matrix,  $U$  has only a few columns, and  $D$  is a non-singular matrix. If  $M_{\text{sp}}$  is positive definite, then we can solve (25.13) by solving a slightly larger but sparse linear system as follows. Let  $\lambda = DU^T\Delta y$ . It is easy to show that (25.13) is equivalent to the following linear system:

$$\begin{bmatrix} M_{\text{sp}} & A^u & U \\ (A^u)^T & 0 & 0 \\ U^T & 0 & -D^{-1} \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta x^u \\ \lambda \end{bmatrix} = \begin{bmatrix} h \\ R_d^u \\ 0 \end{bmatrix}. \quad (25.22)$$

We can use the same method described in Sect. 25.5.5 to solve (25.22).

### 25.5.8 User Supplied Routine to Compute Schur Complement Matrix

The current version of SDPT3 allows the user to supply specialized routines to compute the Schur complement matrices  $M_j^s$  corresponding to the semidefinite blocks.

The specialized routine to compute  $M_j^s$  should have first line that look like:

```
functionschurmat = schurfun_jth(U,V,schurfun_jth_par);
```

where the input arguments  $U$  and  $V$  should correspond to  $x_j^s$  and  $(z_j^s)^{-1}$  if the HKM direction is used; and they should correspond to the NT scaling matrix  $w_j^s$  if the NT direction is used. The third optional argument `schurfun_jth_par` can be a structure array that stores the parameters needed inside the function `schurfun_jth.m`.

The user can tell SDPT3 to use the specialized routine by setting the  $L \times 1$  cell array `OPTIONS.schurfun` as follows: set `OPTIONS.schurfun{j} = schurfun_jth` if  $M_j^s$  is to be computed by the specialized routine coded in the function `schurfun_jth.m`; otherwise set `OPTIONS.schurfun{j} = []`. If the function `schurfun_jth.m` requires some parameters, then the  $L \times 1$  cell array `OPTIONS.schurfun_par` must also be set correspondingly as follows: set `OPTIONS.schurfun_par{j} = schurfun_jth_par`; otherwise set `OPTIONS.schurfun_par{j} = []`.

Below is an example on we how use the specialized routine `mcpeschur.m` in the subdirectory `Examples` to compute the Schur complement matrix when solving the SDP problem `mcp250-1.dat-s`.

```
>> [blk,At,C,b] = read_sdpa('./sdplib/mcp250-1.dat-s');
>> OPTIONS.schurfun{1} = 'mcpeschur';
>> [obj,X,y,Z] = sqlp(blk,At,C,b,OPTIONS);
```

In the above example, there is no need to set the cell array `OPTIONS.schurfun_par` since the function `mcpeschur.m` does not need any additional parameters.

### 25.5.9 Step-Length Computation

Once a direction  $\Delta x$  ( $\Delta z$ ) is computed, a full step will not be allowed if  $x + \Delta x$  ( $z + \Delta z$ ) violates the conic constraints. Thus, the next iterate must take the form  $x + \alpha\Delta x$  ( $z + \beta\Delta z$ ) for an appropriate choice of the step-length  $\alpha$  ( $\beta$ ). In the interest of saving space, we refer the reader to [28, 33] for a detailed description of the efficient computation of the maximum allowed step-length  $\alpha_{\max}$  ( $\beta_{\max}$ ) that can be taken for  $\Delta x$  ( $\Delta z$ ) without violating the conic constraints. In our implementation, we take  $\alpha = \min\{1, \bar{\gamma}\alpha_{\max}\}$  ( $\beta = \min\{1, \bar{\gamma}\beta_{\max}\}$ ), where  $\bar{\gamma}$  (known as the step-length parameter) is typically chosen to be a number slightly smaller than 1, say 0.98, to ensure that the next iterate stays strictly in the interior of all the cones.

## 25.6 Homogeneous Self-Dual Model

In the current version of SDPT3, we have implemented algorithms analogous to Algorithm IPC in Appendix A to solve the following 3-parameter homogeneous self-dual (HSD) model [38] of  $(P)$  and  $(D)$  when the problems have no logarithmic terms or unrestricted variables:

$$(P_H) \quad \begin{aligned} & \min \bar{\alpha}\theta \\ \text{s.t. } & \left[ \begin{array}{ccc|c} 0 & -\mathcal{A} & b & -\bar{b} \\ \mathcal{A}^T & 0 & -c & \bar{c} \\ -b^T & c^T & 0 & -\bar{g} \\ \bar{b}^T & -\bar{c}^T & \bar{g} & 0 \end{array} \right] \begin{bmatrix} y \\ x \\ \tau \\ \theta \end{bmatrix} + \begin{bmatrix} 0 \\ z \\ \kappa \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \bar{\alpha} \end{bmatrix} \end{aligned}$$

$$x \in K, z \in K^*, \tau, \kappa \geq 0, y \in R^m, \theta \in R, \quad (25.23)$$

where for a given  $(x_0, y_0, z_0, \tau_0, \kappa_0, \theta_0)$  such that  $x_0 \in \text{int}(K)$ ,  $z_0 \in \text{int}(K^*)$ ,  $\tau_0, \kappa_0, \theta_0 > 0$ ,  $\bar{\alpha} = (\langle x_0, z_0 \rangle + \tau_0 \kappa_0) / \theta_0$ ,

$$\bar{b} = \frac{1}{\theta_0}(b\tau_0 - Ax_0), \bar{c} = \frac{1}{\theta_0}(c\tau_0 - A^T y_0 - z_0), \bar{g} = \frac{1}{\theta_0}(\langle c, x_0 \rangle - b^T y_0 + \kappa_0).$$

For the (self-dual) model  $(P_H)$ , we have implemented an algorithm that is analogous to Algorithm IPC in the main program `HSDsqlp.m`. At each iteration of the algorithm, the computation of the search direction is very similar to the case for

the problems  $(P)$  and  $(D)$ . For the benefit of readers who are interested in the implementation, here we shall outline the computation of the HKM-like search direction for  $(P_H)$ . (It is only HKM-like because, to keep corresponding iterates in the problem and its dual, it is necessary to mix the HKM and dual HKM directions suitably.) Let

$$\widehat{\mathcal{A}} = [\mathcal{A}; -c^T; \bar{c}^T], \quad \widehat{y} = [y; \tau; \theta], \quad \widehat{B} = \begin{bmatrix} 0 & -b & \bar{b} \\ b^T & 0 & \bar{g} \\ -\bar{b}^T & -\bar{g} & 0 \end{bmatrix}. \quad (25.24)$$

The HKM-like search direction for  $(P_H)$  is the solution of the following linear system of equations:

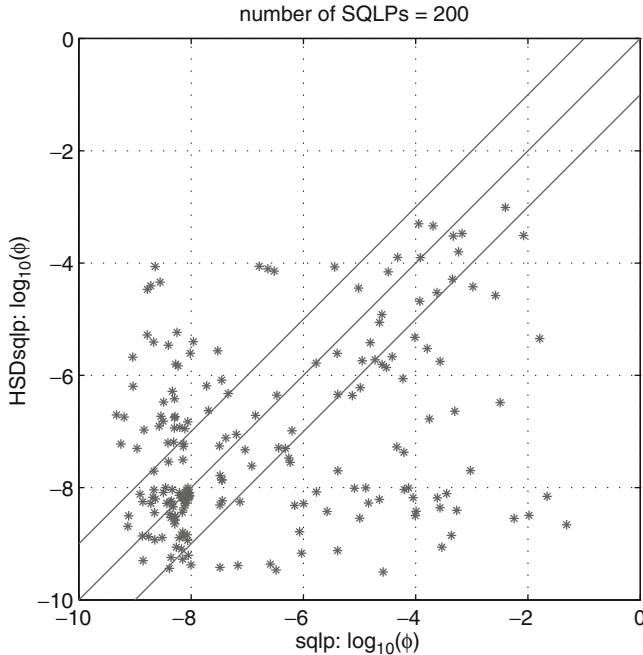
$$\begin{aligned} \widehat{\mathcal{A}}\Delta x + \widehat{B}\Delta\widehat{y} - [0; \Delta\kappa; 0] &= \widehat{R}_p := [0; \kappa; -\bar{\alpha}] - \widehat{\mathcal{A}}x - \widehat{B}\widehat{y} \\ \widehat{\mathcal{A}}^T\Delta\widehat{y} + \Delta z &= R_d := -\widehat{\mathcal{A}}^T\widehat{y} - z \\ \Delta x + H(\Delta z) &= R_c \\ \Delta\kappa + (\kappa/\tau)\Delta\tau &= R_t := \widehat{\mu}/\tau - \kappa, \end{aligned} \quad (25.25)$$

where  $\widehat{\mu} = (\langle x, z \rangle + \tau\kappa)/(n+1)$ . In (25.25), the operator  $H$  is understood to be acting on the individual blocks of  $\Delta z$  as in (25.11) and  $R_c = [R_c^s; R_c^q; R_c^l]$ . From here, the equation analogous to the Schur complement equation in (25.13) without  $A^u$  is given as follows:

$$(\widehat{\mathcal{A}}H\widehat{\mathcal{A}}^T + \widehat{B} + \text{diag}([0; \kappa/\tau; 0]))\Delta\widehat{y} = \widehat{R}_p + \widehat{\mathcal{A}}(H(R_d) - R_c) + [0; R_t; 0]. \quad (25.26)$$

The main difference between (25.26) and (25.13) without  $A^u$  lies in the term  $\widehat{B}$ , which is an skew-symmetric matrix with rank at most four, generally containing dense columns. To efficiently solve (25.26), we use the dense-column handling technique described in Sect. 25.5.7 to solve the linear system.

We have found that the solution obtained by the solver `HSDsqlp.m` based on the HSD model  $(P_H)$  is generally more accurate than the one obtained by `sqlp.m` based on  $(P)$  and  $(D)$  when the SQLP problem is feasible but either the primal or dual feasible region has an empty interior. In Fig. 25.1, we plotted the accuracies attained by `HSDsqlp.m` against those attained by `sqlp.m` for about 200 SQLPs whose primal or dual feasible regions are very likely to have empty interiors (based on the fact that the value  $\max\{g_P, g_D\}$  described in Sect. 25.7 are larger than  $10^{12}$ ). We may observe from the scattered plot in Fig. 25.1 that `HSDsqlp.m` tends to give more accurate solutions compared to `sqlp.m`.



**Fig. 25.1** Accuracy attained by HSDsqlp.m versus that attained by sqlp.m for SQLPs whose primal or dual feasible regions have empty interiors

## 25.7 Detecting Feasible Regions with Empty Interiors

In this section, we only consider the case where there are no unrestricted variables  $x^u$  in  $(P)$ . Let  $d = (A, b, c)$  be the SQLP data associated with  $(P)$  and  $(D)$  and  $F_P(d)$  and  $F_D(d)$  be their respective feasible regions. It is often of interest to know whether the interiors,  $F_P^\circ(d)$  and  $F_D^\circ(d)$ , are empty, and how “thick” these regions are. A natural quantitative measure of the “thickness” of  $F_P(d)$  and  $F_D(d)$  is the concept of primal and dual distances to infeasibility defined by Renegar [25]:

$$\begin{aligned}\rho_P(d) &= \inf\{\|\Delta d\| : F_P(d + \Delta d) = \emptyset\}, \\ \rho_D(d) &= \inf\{\|\Delta d\| : F_D(d + \Delta d) = \emptyset\}.\end{aligned}$$

With an appropriately chosen norm in the above definitions, the computation of  $\rho_D(d)$  amounts to solving an SQLP problem with roughly the same dimension and structure as the original primal instance. Unfortunately, the computation of  $\rho_P(d)$  is extremely costly, requiring the solutions of  $2m$  SQLP problems each with roughly the same dimension and structure as the original dual instance; see [9].

However, in practice, one is typically only interested in the magnitudes of  $\rho_P(d)$  and  $\rho_D(d)$  rather than the exact values. It turns out that the following geometric

measures proposed by Freund [8] usually give enough information about the magnitudes of  $\rho_P(d)$  and  $\rho_D(d)$ :

$$g_P(d) = \inf \left\{ \max \left\{ \|x\|, \frac{\|x\|}{r}, \frac{1}{r} \right\} : \mathcal{A}(x) = b, x - re \in K \right\} \quad (25.27)$$

$$g_D(d) = \inf \left\{ \max \left\{ \|z\|, \frac{\|z\|}{r}, \frac{1}{r} \right\} : \mathcal{A}^T(y) + z = c, z - re \in K \right\}, \quad (25.28)$$

where  $e$  is the unit element in  $K$  and  $\|\cdot\|$  is an appropriately chosen norm. Note that  $g_P(d)$  is smaller to the extent that there is a primal feasible solution that is not too large and that is not too close to the boundary of  $K$ . Similar interpretation holds also for  $g_D(d)$ . It can be shown that  $g_P(d) = \infty \Leftrightarrow \rho_P(d) = 0$  and  $g_D(d) = \infty \Leftrightarrow \rho_D(d) = 0$ .

Freund [8] showed that  $g_P(d)$  ( $g_D(d)$ ) can be computed at the cost of solving a single SQLP problem with roughly the same dimension and structure as the original primal (dual) instance. In the current release of SDPT3, we include the following m-files to compute  $g_P(d)$  and  $g_D(d)$ :

```
gp = gpcomp(blk, At, C, b);    gd = gdcomp(blk, At, C, b);
```

The above routines are based on the standard SQLP formulations of (25.27) and (25.28) derived in [9].

Let  $(x^*, y^*, z^*)$  be an optimal solution to  $(P)$  and  $(D)$ , respectively. The geometric measures  $g_P, g_D$  and  $\max\{\|x^*\|, \|z^*\|\}$  for the test problems considered in Sect. 25.8 are listed in Table 25.1. In the table, we declare that  $g_P$  ( $g_D$ ) is equal to  $\infty$  if the computed number is larger than  $10^{12}$ . We have observed that there is strong correlation between  $g := \max\{g_P, g_D\} \max\{\|x^*\|, \|z^*\|\}$  and the accuracy  $\phi$  one can obtain when solving  $(P)$  and  $(D)$ . In Fig. 25.2, we plotted the measure  $g$  and the accuracy  $\phi$  attainable by `sqlp.m` and `HSDsqlp.m` for about 420 SQLP problems. The sample correlation between  $\log_{10}(\phi)$  with  $\phi$  obtained by `sqlp.m` and  $\log_{10}(g)$  for about 170 SQLPs with finite  $g$  has a correlation coefficient of 0.75. Coincidentally, we have the same correlation coefficient for the case where  $\phi$  is obtained by `HSDsqlp.m`.

## 25.8 Computational Results

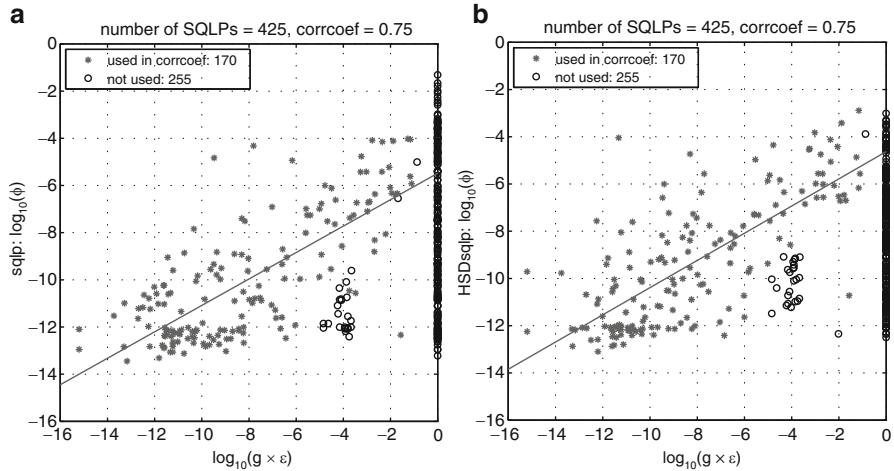
Here we describe the results of our computational testing of SDPT3-4.0 using the default parameters, on problems from the following sources:

1. SDPLIB collection of Borchers, available at  
<http://www.nmt.edu/~borchers/sdplib.html>.
2. DIMACS Challenge test problems, available at  
<http://dimacs.rutgers.edu/Challenges/Seventh/Instances/>.

**Table 25.1** Performance of `sdpt3.m`. In the table, err = [pinfeas, dinfeas, relgap, relgap2], where `relgap2` is the same as `relgap` but with the numerator replaced by  $\|c, x^*\) - b^T y\|$ , and normXZ =  $\max(\|x^*\|, \|z^*\|)$ . We declare that  $g_P(g_D)$  is  $\infty$  if the computed number is larger than  $10^{12}$

Problem	m   n_s   n_q   n_u	it.	primal obj   dual obj	err	Time	$g_P(g_D)$   normXZ
arch0	174   161; : 174;	26	-5.66517270-1   -5.66517274-1	5.4-9   1.6-11   1.9-9   1.8-9	0.3	1.974   2.036   5.92
arch2	174   161; : 174;	24	-6.71515400-1   -6.71515409-1	4.2-10   3.0-11   4.0-9   3.7-9	0.3	1.974   2.016   5.72
arch4	174   161; : 174;	22	-9.72627409-1   -9.72627419-1	7.3-10   1.9-11   3.8-9   3.7-9	0.3	1.974   1.966   9.12
arch8	174   161; : 174;	23	-7.056980020   -7.056980040	9.7-9   2.9-11   1.5-9   1.2-9	0.3	1.974   1.836   6.33
control1	21   15; :	17	-1.778462681   -1.778462671	3.1-9   2.7-11   1.8-9   3.2-10	0.0	0.314   4.983   5.75
control2	66   30; ;	21	-8.300000390   -8.299999990	7.1-9   3.0-11   9.4-11   2.3-8	0.1	3.035   1.474   6.55
control3	136   45; ;	22	-1.363326471   -1.363326721	1.9-7   8.7-11   1.0-7   8.9-8	0.2	7.675   3.164   2.16
control4	23   60; ;	21	-1.979423251   -1.979423081	1.9-7   1.5-10   1.7-8   4.1-8	0.4	1.346   4.924   3.86
control5	351   75; ;	23	-1.688359361   -1.688360101	4.3-7   1.6-10   2.5-7   2.1-7	11	2.026   6.224   4.96
control6	496   90; ;	21	-3.730436481   -3.730442731	2.2-7   5.8-10   8.1-7   8.3-7	22	3.126   9.214   1.47
control7	666   105; ;	22	-2.062505811   -2.062507781	5.9-8   4.9-10   4.7-7   4.7-7	43	4.096   1.155   1.27
control8	861   120; ;	23	-2.028634781   -2.028636531	2.0-7   4.8-10   4.0-7   4.2-7	1.23	5.536   1.405   1.37
control9	1081   135; ;	23	-1.467541571   -1.467542841	2.7-7   4.7-10   4.6-7   4.2-7	2.24	6.986   1.725   1.37
control10	1326   150; ;	25	-3.853286871   -3.853305821	5.0-7   1.4-9   2.3-6   2.4-6	45	8.326   2.005   3.77
control11	1596   165; ;	24	-3.195860901   -3.195868621	5.9-7   1.3-9   8.7-7   1.2-6	1.07	1.027   2.315   3.47
gpp100	101   100; ;	14	4.494354791   4.494354891	2.7-10   6.2-11   9.8-9   1.1-8	0.0	$\infty$   1.882   6.24

gpp124-1	125   124;;;	17   7.34307525 0   7.34307571 0	3.6-11  7.1-12  7.6-9  2.9-8	01	$\infty$	1.92   2   1.75
gpp124-2	125   124;;;	15   4.68622933 1   4.68622939 1	1.1-10  2.2-11  6.1-9  6.3-9	01	$\infty$	2.37   2   1.1.5
gpp124-3	125   124;;;	14   1.53014123 2   1.53014124 2	4.2-10  8.5-11  4.1-9  5.1-9	01	$\infty$	2.83   2   6.9.4
gpp124-4	125   124;;;	15   4.18987595 2   4.18987610 2	5.6-10  7.1-11  6.3-10  1.8-8	01	$\infty$	3.48   2   2.9.5
gpp250-1	251   250;;;	18   1.54449168 1   1.54449168 1	1.3-12  1.5-12  2.9-9  8.7-11	02	$\infty$	4.01   2   1.0.6
gpp250-2	251   250;;;	15   8.18689562 1   8.18689574 1	1.3-10  2.6-11  1.2-9  7.4-9	02	$\infty$	4.76   2   1.6.5
gpp250-3	251   250;;;	15   3.03539317 2   3.03539320 2	3.3-10  6.7-11  2.1-9  5.2-9	02	$\infty$	5.91   2   1.9.5
gpp250-4	251   250;;;	14   7.47328306 2   7.47328305 2	2.2-10  5.3-11  4.6-9  5.8-10	02	$\infty$	7.20   2   4.0.5
gpp500-1	501   500;;;	20   2.53205508 1   2.53205436 1	1.6-12  2.6-12  1.4-7  1.4-7	12	$\infty$	7.88   2   1.3.7
gpp500-2	501   500;;;	19   1.56060387 2   1.56060387 2	3.1-12  6.9-12  6.4-10  7.7-11	11	$\infty$	9.57   2   1.6.6
gpp500-3	501   500;;;	16   5.13017610 2   5.13017602 2	4.3-12  2.0-12  7.7-9  7.5-9	10	$\infty$	1.17   3   1.0.6
gpp500-4	501   500;;;	17   1.56701879 3   1.56701879 3	8.8-12  3.8-12  9.2-10  8.2-10	10	$\infty$	1.50   3   7.7.5

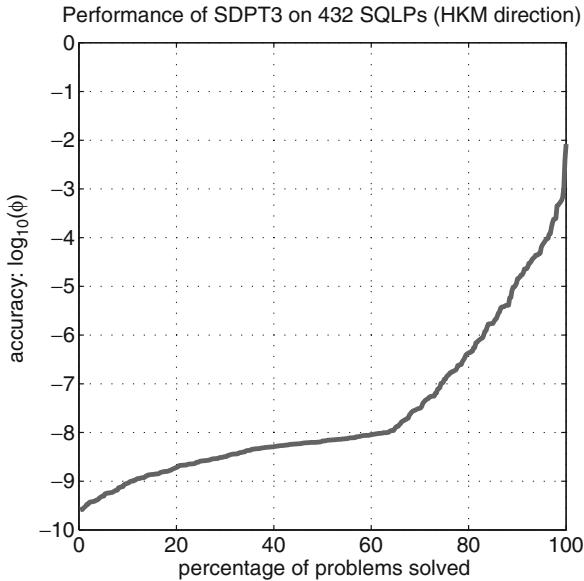


**Fig. 25.2** The data points corresponding to “o” are not used in the calculation of the correlation coefficient because either  $1/\max\{g_P, g_D\} = 0$  or because it is not computed to sufficient accuracy to determine whether the number is in fact 0. On the  $x$ -axis,  $\varepsilon = 2.2 \times 10^{-16}$ , and  $g = \max\{g_P, g_D\} \max\{\|x^*\|, \|z^*\|\}$

3. Sparse SDPs from structural optimization, available at <http://www2.am.uni-erlangen.de/~kocvara/pennon/problems.html>.
4. Sparse SDP collection of Hans Mittelmann, available at <ftp://plato.asu.edu/pub/sdp/>.
5. SDPs from electronic structure calculations, available at <http://www.cims.nyu.edu/~mituhiro/software.html>.
6. SDPs from polynomial optimizations [15, 16].
7. SOCP problems generated by the MATLAB FIR filter toolbox, available at <http://www.csee.umbc.edu/~dschol2/opt.html>.
8. SDPs from polynomial optimizations [36].

Our results were obtained on an Intel Xeon 3.0 GHz PC with 4G of memory running Linux and MATLAB 7.6. Figure 25.3 shows the performance of the hybrid solver `sdpt3.m` on a total of about 430 SQLP problems. It shows that `sdpt3.m` was able to solve more than 80% of the problems to an accuracy of at least  $10^{-6}$  in the measure  $\phi$  defined in (25.6).

Detailed information such as primal and dual objective values, error measures such as `pinfeas`, `dinfeas`, `relgap` as defined in (25.6), the geometric measures  $g_P, g_D, \max\{\|x^*\|, \|z^*\|\}$ , and the CPU time taken for each problem can be found in Table 25.1. Note that because of space limitation, we only list the results for a subset of the 430 instances we have tested. The complete set of results can be found from the web site: <http://www.math.nus.edu.sg/~mattohkc/sdpt3.html>.



**Fig. 25.3**  $\phi$  is defined as in (25.6)

## 25.9 Future Work

In a future release of SDPT3, we plan to include the following additional features.

- (a) An implementation of a path-following algorithm which targets high accuracy solutions at the expense of significantly longer computing time per iteration. The key idea would be to compute the search direction at each iteration based on a reduced augmented equation as formulated in [32] and [4] that has at most twice the dimension of the Schur complement equation.
- (b) The flexibility of inputting  $\mathcal{A}$  in (25.1) as an user supplied function instead of the current requirement of being the matrix representation of  $\mathcal{A}$ . Such a flexibility is highly useful for certain SDPs such as the example considered in Sect. 25.4.4.
- (c) Ability to handle complex data corresponding to the second-order cone blocks.

## Appendix: A Primal-Dual Infeasible-Interior-Point Algorithm

Here we give a **pseudo-code** for the algorithm we implemented. Note that this description makes references to earlier sections where details related to the algorithm are explained.

### Algorithm IPC

Suppose we are given an initial iterate  $(x^0, y^0, z^0)$  with  $x^0, z^0$  strictly satisfying all the conic constraints. Decide on the type of search direction to use. Set  $\bar{\gamma}^0 = 0.9$ .

**For**  $k = 0, 1, \dots$

(Let the current and the next iterate be  $(x, y, z)$  and  $(x^+, y^+, z^+)$  respectively. Also, let the current and the next step-length parameter be denoted by  $\bar{\gamma}$  and  $\bar{\gamma}^+$  respectively.)

- Compute  $\mu(x, z)$  defined in (25.5), and the accuracy measure  $\phi$  defined in (25.6). Stop the iteration if  $\phi$  is sufficiently small.
- (Predictor step) Solve the linear system (25.13) with  $\sigma = 0$  in the right-side vector (25.15). Denote the solution of (25.11) by  $(\delta x, \delta y, \delta z)$ . Let  $\alpha_p$  and  $\beta_p$  be the step-lengths defined as in Sect. 25.5.9 with  $\Delta x, \Delta z$  replaced by  $\delta x, \delta z$ , respectively.
- Take  $\sigma$  to be

$$\sigma = \min \left( 1, \left[ \frac{\mu(x + \alpha_p \delta x, z + \beta_p \delta z)}{\mu(x, z)} \right]^e \right),$$

where the exponent  $e$  is chosen as follows:

$$e = \begin{cases} \max[1, 3 \min(\alpha_p, \beta_p)^2] & \text{if } \mu(x, z) > 10^{-6}, \\ 1 & \text{if } \mu(x, z) \leq 10^{-6}. \end{cases}$$

- (Corrector step) Solve the linear system (25.13) with  $R_c$  in the right-hand side vector (25.15) replaced by

$$\widehat{R}_c^\tau = R_c^\tau - \text{Mehrotra-corrector term generated from } \delta x^\tau \text{ and } \delta z^\tau, \quad \tau \in \{s, q, l\}.$$

Denote the solution of (25.11) by  $(\Delta x, \Delta y, \Delta z)$ .

- Update  $(x, y, z)$  to  $(x^+, y^+, z^+)$  by

$$x^+ = x + \alpha \Delta x, \quad y^+ = y + \beta \Delta y, \quad z^+ = z + \beta \Delta z,$$

where  $\alpha$  and  $\beta$  are computed as in Sect. 25.5.9 with  $\bar{\gamma}$  chosen to be  $\bar{\gamma} = 0.9 + 0.09 \min(\alpha_p, \beta_p)$ .

- Update the step-length parameter by  $\bar{\gamma}^+ = 0.9 + 0.09 \min(\alpha, \beta)$ .

## References

1. Alizadeh, F., Haeberly, J.-P.A., Overton, M.L.: Primal-dual interior-point methods for semidefinite programming: convergence results, stability and numerical results. SIAM J. Optimization **8**, 746–768 (1998)

2. Borchers, B.: SDPLIB 1.2, a library of semidefinite programming test problems. *Optimization Methods and Software* **11 & 12**, 683–690 (1999)
3. Boyd, S., El Ghaoui, L., Feron, E., Balakrishnan, V.: *Linear Matrix Inequalities in System and Control Theory*. SIAM, Philadelphia (1994)
4. Cai, Z., Toh, K.C.: Solving second order cone programming via the augmented systems. *SIAM J. Optimization* **17**, 711–737 (2006)
5. Davis, T.A.: CHOLMOD Version 1.0 User Guide. Technical Report, University of Florida, Gainesville, Florida (2005). <http://www.cise.ufl.edu/research/sparse/cholmod>
6. Davis, T.A.: UMFPACK Version 4.6 User Guide. Technical Report, University of Florida, Gainesville, Florida (2002). <http://www.cise.ufl.edu/research/sparse/umfpack>
7. Duff, I.S.: MA57 – A new code for the solution of sparse symmetric definite and indefinite systems. Technical Report RAL-TR-2002-024, Rutherford Appleton Laboratory, Oxford (2002)
8. Freund, R.M.: Complexity of convex optimization using geometry-based measures and a reference point, *Mathematical Programming* **99**, 197–221 (2004)
9. Freund, R.M., Ordóñez, F., Toh, K.C., Behavioral Measures and their Correlation with IPM Iteration Counts on Semi-Definite Programming Problems. *Mathematical Programming* **109**, 445–475 (2007)
10. Freund, R.W., Nachtigal, N.M.: A new Krylov-subspace method for symmetric indefinite linear systems. In: Ames, W.F. (ed) *Proceedings of the 14th IMACS World Congress on Computational and Applied Mathematics*, Atlanta (1994)
11. Fujisawa, K., Kojima, M., Nakata, K.: Exploiting sparsity in primal-dual interior-point method for semidefinite programming. *Mathematical Programming* **79**, 235–253 (1997)
12. Grant, M., Boyd, S.: Graph implementations for nonsmooth convex programs. In: Blondel, V., Boyd, S., Kimura, H. (eds.) *Recent Advances in Learning and Control (a tribute to M. Vidyasagar)*. Lecture Notes in Control and Information Sciences. Springer, Berlin Heidelberg New York (2008). Software is available at <http://cvxr.com/cvx/>
13. Harwell Subroutine Library: <http://www.cse.clrc.ac.uk/Activity/HSL> (1963)
14. Helmburg, C., Rendl, F., Vanderbei, R., Wolkowicz, H.: An interior-point method for semidefinite programming. *SIAM Journal on Optimization* **6**, 342–361 (1996)
15. Henrion, D., Lasserre, J.B.: GloptiPoly: Global optimization over polynomials with Matlab and SeDuMi. *ACM Transactions on Mathematical Software* **29**, 165–194 (2003)
16. Henrion, D., Lasserre, J.B., Löfberg, J.: GloptiPoly 3: moments, optimization and semidefinite programming. *Optimization Methods and Software* **24**, 761–779 (2009)
17. Kojima, M., Shindoh, S., Hara, S.: Interior-point methods for the monotone linear complementarity problem in symmetric matrices. *SIAM J. Optimization* **7**, 86–125 (1997)
18. Liu, J.W., Ng, E.G., Peyton, B.W.: On finding supernodes for sparse matrix computations. *SIAM J. Matrix Anal. Appl.* **1**, 242–252 (1993)
19. Löfberg, J.: A Toolbox for Modeling and Optimization in MATLAB. In: *Proceedings of the CACSD Conference*, Taipei, Taiwan (2004). <http://control.ee.ethz.ch/~joloef/yalmip.php>
20. Mehrotra, S.: On the implementation of a primal-dual interior point method. *SIAM J. Optimization* **2**, 575–601 (1992)
21. Monteiro, R.D.C.: Primal-dual path-following algorithms for semidefinite programming. *SIAM J. Optimization* **7**, 663–678 (1997)
22. Monteiro, R.D.C., Tsuchiya, T.: Polynomial convergence of primal-dual algorithms for the second-order cone program based on the MZ-family of directions. *Mathematical Programming* **88**, 61–83 (2000)
23. Nesterov, Y., Todd, M.J.: Self-scaled barriers and interior-point methods in convex programming. *Mathematics of Operations Research* **22**, 1–42 (1997)
24. Pataki, G., Schmieta, S.: The DIMACS library of mixed semidefinite-quadratic-linear programs. <http://dimacs.rutgers.edu/Challenges/Seventh/Instances> (1999)
25. Renegar, J.: Linear programming, complexity theory, and elementary functional analysis. *Mathematical Programming* **70**, 279–351 (1995)
26. Sturm, J.F.: Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software* **11 & 12**, 625–653 (1999)

27. Todd, M.J., Toh, K.C., Tütüncü, R.H.: On the Nesterov-Todd direction in semidefinite programming. *SIAM J. Optimization*, **8**, 769–796 (1998)
28. Toh, K.C.: A note on the calculation of step-lengths in interior-point methods for semidefinite programming. *Computational Optimization and Applications* **21**, 301–310 (2002)
29. Toh, K.C.: Some new search directions for primal-dual interior point methods in semidefinite programming. *SIAM J. Optimization* **11**, 223–242 (2000)
30. Toh, K.C.: Primal-dual path-following algorithms for determinant maximization problems with linear matrix inequalities. *Computational Optimization and Applications* **14**, 309–330 (1999)
31. Toh, K.C., Todd, M.J., Tütüncü, R.H.: SDPT3 — a Matlab software package for semidefinite programming. *Optimization Methods and Software* **11 & 12**, 545–581 (1999)
32. Toh, K.C.: Solving large scale semidefinite programs via an iterative solver on the augmented systems. *SIAM J. Optim.* **14**, 670–698 (2004)
33. Tütüncü, R.H., Toh, K.C., Todd, M.J., Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming* **95**, 189–217 (2003)
34. Tsuchiya, T.: A convergence analysis of the scaling-invariant primal-dual path-following algorithms for second-order cone programming. *Optimization Methods and Software* **11 & 12**, 141–182 (1999)
35. Vandenberghe, L., Boyd, S., Wu, S.-P.: Determinant maximization with linear matrix inequalities. *SIAM J. Matrix Analysis and Applications* **19**, 499–533 (1998)
36. Waki, H., Kim, S., Kojima, M., Muramatsu, M., Sugimoto, H.: SparsePOP: a sparse semidefinite programming relaxation of polynomial optimization problems. *ACM Transactions on Mathematical Software* **35**, article 15 (2008)
37. Yamashita, M., Fujisawa, K., Kojima, M.: Implementation and evaluation of SDPA 6.0 (SemiDefinite Programming Algorithm 6.0). *Optimization Methods and Software* **18**, 491–505 (2003)
38. Ye, Y., Todd, M.J., Mizuno, S.: An  $O(\sqrt{n}L)$ -iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research* **19**, 53–67 (1994)

# Chapter 26

## PENNON: Software for Linear and Nonlinear Matrix Inequalities

Michal Kocvara and Michael Stingl

### 26.1 Introduction

The goal of this chapter is to present an overview of the software collection for the solution of linear and nonlinear semidefinite optimization problems PENNON. In the first part we present theoretical and practical details of the underlying algorithm and several implementation issues. In the second part we introduce the particular codes PENSDP, PENBMI and PENNON, focus on some specific features of these codes and show how they can be used for the solution of selected problems.

We use standard notation:  $\mathbb{S}^m$  is the space of real symmetric matrices of dimension  $m \times m$  and  $\mathbb{S}_+^m$  the space of positive semidefinite matrices from  $\mathbb{S}^m$ . The inner product on  $\mathbb{S}^m$  is defined by  $\langle A, B \rangle_{\mathbb{S}^m} := \text{trace}(AB)$ . Notation  $A \preccurlyeq B$  for  $A, B \in \mathbb{S}^m$  means that the matrix  $B - A$  is positive semidefinite. The norm  $\|\cdot\|$  is always the  $\ell_2$  norm in case of vectors and the spectral norm in case of matrices, unless stated otherwise. Finally, for  $\Phi : \mathbb{S}^m \rightarrow \mathbb{S}^m$  and  $X, Y \in \mathbb{S}^m$ ,  $D\Phi(X)[Y]$  denotes the directional derivative of  $\Phi$  with respect to  $X$  in direction  $Y$ .

---

M. Kocvara (✉)

School of Mathematics, University of Birmingham, Birmingham B15 2TT, UK

Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic,  
Pod vodárenskou veží 4,

18208 Praha 8, Czech Republic

e-mail: [kocvara@maths.bham.ac.uk](mailto:kocvara@maths.bham.ac.uk)

M. Stingl

Institute of Applied Mathematics, University of Erlangen-Nuremberg,  
Martensstrasse 3, 91058 Erlangen, Germany

e-mail: [stingl@am.uni-erlangen.de](mailto:stingl@am.uni-erlangen.de)

## 26.2 The Main Algorithm

### 26.2.1 Problem Formulation

The nonlinear semidefinite problems can be written in several different ways. In this section, for the sake of simplicity, we will use the following formulation:

$$\begin{aligned} & \min_{x \in \mathbb{R}^n} f(x) \\ & \text{subject to} \\ & \quad \mathcal{A}(x) \leq 0. \end{aligned} \tag{26.1}$$

Here  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  and  $\mathcal{A} : \mathbb{R}^n \rightarrow \mathbb{S}^m$  are twice continuously differentiable mappings.

Later in sections on linear SDP, BMI and nonlinear SDP, we will give more specific formulations of the problem. However, the algorithm and theory described in this section applies, with some exceptions discussed later, to all these specific formulations.

### 26.2.2 The Algorithm

The basic algorithm used in this article is based on the nonlinear rescaling method of R. Polyak [30] and was described in detail in [16] and [31]. Here we briefly recall it and stress points that will be needed in the rest of the chapter.

The algorithm is based on the choice of a smooth penalty/barrier function  $\Phi_p : \mathbb{S}^m \rightarrow \mathbb{S}^m$  that satisfies a number of assumptions (see [16, 31]) guaranteeing, in particular, that for any  $p > 0$

$$\mathcal{A}(x) \leq 0 \iff \Phi_p(\mathcal{A}(x)) \leq 0$$

for (at least) all  $x$  such that  $\mathcal{A}(x) \leq 0$ . Thus for any  $p > 0$ , problem (26.1) has the same solution as the following “augmented” problem

$$\begin{aligned} & \min_{x \in \mathbb{R}^n} f(x) \\ & \text{subject to} \\ & \quad \Phi_p(\mathcal{A}(x)) \leq 0. \end{aligned} \tag{26.2}$$

The Lagrangian of (26.2) can be viewed as a (generalized) augmented Lagrangian of (26.1):

$$F(x, U, p) = f(x) + \langle U, \Phi_p(\mathcal{A}(x)) \rangle_{\mathbb{S}^m}; \tag{26.3}$$

here  $U \in \mathbb{S}_+^m$  is a Lagrangian multiplier associated with the inequality constraint.

The algorithm below can be seen as a generalization of the Augmented Lagrangian method.

**Algorithm 26.1.** Let  $x^1$  and  $U^1$  be given. Let  $p^1 > 0$ . For  $k = 1, 2, \dots$  repeat until a stopping criterion is reached:

1.  $x^{k+1} = \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} F(x, U^k, p^k).$
2.  $U^{k+1} = D\Phi_p(\mathcal{A}(x^{k+1}))[U^k].$
3.  $p^{k+1} \leq p^k.$

Details of the algorithm, the choice of the penalty function  $\Phi_p$ , the choice of initial values of  $x, U$  and  $p$ , the approximate minimization in Step (i) and the update formulas, will be discussed in subsequent sections. The next section concerns the overview of theoretical properties of the algorithm.

### 26.2.3 Convergence Theory Overview

Throughout this section we make the following assumptions on problem (26.1):

- (A1)  $x^* = \arg \min \{f(x) | x \in \Omega\}$  exists, where  $\Omega = \{x \in \mathbb{R}^n | \mathcal{A}(x) \leq 0\}$ .
- (A2) The Karush–Kuhn–Tucker necessary optimality conditions hold in  $x^*$ , i.e., there exists  $U^* \in \mathbb{S}^m$  such that

$$\begin{aligned} f'(x^*) + [\langle U^*, \mathcal{A}_i \rangle]_{i=1}^m &= 0 \\ \langle U^*, \mathcal{A}(x^*) \rangle &= 0 \\ U^* &\geq 0 \\ \mathcal{A}(x^*) &\leq 0, \end{aligned} \tag{26.4}$$

where  $\mathcal{A}_i$  denotes the  $i$ -th partial derivative of  $\mathcal{A}$  at  $x^*$  ( $i = 1, \dots, n$ ). Moreover the strict complementary is satisfied.

- (A3) The nondegeneracy condition holds, i.e., if for  $1 \leq r < m$  the vectors  $s_{m-r+1}, \dots, s_m \in \mathbb{R}^m$  form a basis of the null space of the matrix  $\mathcal{A}(x^*)$ , then the following set of  $n$ -dimensional vectors is linearly independent:

$$v_{i,j} = (s_i^\top \mathcal{A}_1 s_j, \dots, s_i^\top \mathcal{A}_n s_j)^\top, \quad m-r+1 \leq i \leq j \leq m.$$

- (A4) Define  $E_0 = (s_{m-r+1}, \dots, s_m)$ , where  $s_{m-r+1}, \dots, s_m$  are the vectors introduced in assumption (A3). Then the *cone of critical directions* at  $x^*$  is defined as

$$C(x^*) = \left\{ h \in \mathbb{R}^n : \sum_{i=1}^n h_i E_0^\top \mathcal{A}_i E_0 \leq 0, f'(x^*)^\top h = 0 \right\}.$$

With this the following second order sufficient optimality condition is assumed to hold at  $(x^*, U^*)$ : For all  $h \in C(x^*)$  with  $h \neq 0$  the inequality

$$h^\top (L''_{xx}(x^*, U^*) + H(x^*, U^*)) h > 0,$$

is satisfied, where  $L$  is the classic Lagrangian of (26.1) defined as

$$L(x, U) = f(x) + \langle U, \mathcal{A}(x) \rangle,$$

$H(x^*, U^*)$  is defined entry-wise by

$$H(x^*, U^*)_{i,j} = -2 \left\langle U^*, \mathcal{A}_i[\mathcal{A}(x^*)]^\dagger \mathcal{A}_j \right\rangle \quad (26.5)$$

(see, for example, [4, p. 490]) and  $[\mathcal{A}(x^*)]^\dagger$  is the Moore–Penrose inverse of  $\mathcal{A}(x^*)$ .

(A5) Let

$$\mathcal{Q}_p = \{x \in \mathbb{R}^n \mid \mathcal{A}(x) \leq pI_m\}.$$

Then the following growth condition holds:

$$\exists \pi > 0 \text{ and } \tau > 0 \text{ such that } \max \{\|\mathcal{A}(x)\| \mid x \in \mathcal{Q}_\pi\} \leq \tau. \quad (26.6)$$

Using these assumptions the following local convergence result can be established.

**Theorem 26.1.** *Let  $\mathcal{A}(x)$  be twice continuously differentiable and assumptions (A1) to (A5) hold for the pair  $(x^*, U^*)$ . Then there exists a penalty parameter  $p_0 > 0$  large enough and a neighbourhood  $\mathcal{V}$  of  $(x^*, U^*)$  such that for all  $(U, p) \in \mathcal{V}$ :*

(a) *There exists a vector*

$$\hat{x} = \hat{x}(U, p) = \arg \min \{F(x, U, p) \mid x \in \mathbb{R}^n\}$$

*such that  $\nabla_x F(\hat{x}, U, p) = 0$ .*

(b) *For the pair  $\hat{x}$  and  $\widehat{U} = \widehat{U}(U, p) = D\Phi_p(\mathcal{A}(\hat{x}(U, p)))[U]$  the estimate*

$$\max \left\{ \|\hat{x} - x^*\|, \|\widehat{U} - U^*\| \right\} \leq Cp \|U - U^*\| \quad (26.7)$$

*holds, where  $C$  is a constant independent of  $p$ .*

(c)  $\hat{x}(U^*, p) = x^*$  and  $\widehat{U}(U^*, p) = U^*$ .

(d) *The function  $F(x, U, p)$  is strongly convex with respect to  $x$  in a neighborhood of  $\hat{x}(U, p)$ .*

The proof for Theorem 26.1 as well as a precise definition of the neighborhood  $\mathcal{V}$  is given in [31]. A slightly modified version of Theorem 26.1 with a particular choice of the penalty function  $\Phi$  can be found in [21]. An alternative convergence theorem using slightly different assumptions is presented in [29].

An immediate consequence of Theorem 26.1 is that Algorithm 26.1 converges with a linear rate of convergence. If  $p_k \rightarrow 0$  for  $k \rightarrow \infty$  is assumed for the sequence of penalty parameters then the rate of convergence is superlinear.

*Remark 26.1.* (a) Let  $x^+$  be a local minimum of problem (26.1) satisfying assumptions (A2) to (A5) and denote by  $U^+$  the corresponding (unique) optimal multiplier. Assume further that there exists a neighborhood  $S_\nu$  of  $x^+$  such that there is no further first order critical point  $\tilde{x} \neq x^+$  in  $S_\nu(x^+)$ . Then all statements of Theorem 26.1 remain valid, if we replace  $(x^*, U^*)$  by  $(x^+, U^+)$  and the function  $\hat{x}(U, p)$  by

$$\hat{x}_{\text{loc}}(U, p) = \arg \min \{F(x, U, p) | x \in \mathbb{R}^n, x \in S_\nu\}. \quad (26.8)$$

Moreover Theorem 26.1 (d) guarantees that  $F(x, U, p)$  is strongly convex in a neighborhood of  $x^+$  for all  $(U, p) \in \mathcal{V}$ . Consequently any local descent method applied to the problem

$$(i') \quad \text{Find } x^{k+1} \text{ such that } \|\nabla_x F(x, U^k, p^k)\| = 0 \quad (26.9)$$

will automatically find a solution, which satisfies the additional constraint  $x^{k+1} \in S_\nu$  provided it is started with  $x^k$  close enough to  $x^+$ . Moreover, Algorithm 26.1 will converge to the local optimum  $x^+$  (see [31] for more details).

(b) A global convergence result can be found in [31].

#### 26.2.4 Choice of $\Phi_p$

The penalty function  $\Phi_p$  of our choice is defined as follows:

$$\Phi_p(\mathcal{A}(x)) = -p^2(\mathcal{A}(x) - pI)^{-1} - pI. \quad (26.10)$$

The advantage of this choice is that it gives closed formulas for the first and second derivatives of  $\Phi_p$ . Defining

$$\mathcal{Z}(x) = -(\mathcal{A}(x) - pI)^{-1} \quad (26.11)$$

we have (see [16]):

$$\frac{\partial}{\partial x_i} \Phi_p(\mathcal{A}(x)) = p^2 \mathcal{Z}(x) \frac{\partial \mathcal{A}(x)}{\partial x_i} \mathcal{Z}(x) \quad (26.12)$$

$$\begin{aligned} \frac{\partial^2}{\partial x_i \partial x_j} \Phi_p(\mathcal{A}(x)) = p^2 \mathcal{Z}(x) & \left( \frac{\partial \mathcal{A}(x)}{\partial x_i} \mathcal{Z}(x) \frac{\partial \mathcal{A}(x)}{\partial x_j} + \frac{\partial^2 \mathcal{A}(x)}{\partial x_i \partial x_j} \right. \\ & \left. + \frac{\partial \mathcal{A}(x)}{\partial x_j} \mathcal{Z}(x) \frac{\partial \mathcal{A}(x)}{\partial x_i} \right) \mathcal{Z}(x). \end{aligned} \quad (26.13)$$

### 26.2.5 The Modified Newton Method

To solve the (possibly nonconvex) unconstrained minimization problem in Step 1, we use the following modification of the Newton method with line-search:

**Algorithm 26.2.** Given an initial iterate  $x_0$ , repeat for all  $k = 0, 1, 2, \dots$  until a stopping criterion is reached:

1. Compute the gradient  $g_k$  and Hessian  $H_k$  of  $F$  at  $x_k$ .
2. Try to factorize  $H_k$  by Cholesky decomposition. If  $H_k$  is factorizable, set  $\widehat{H} = H_k$  and go to Step 4.
3. Compute  $\beta \in [-\lambda_{\min}, -2\lambda_{\min}]$ , where  $\lambda_{\min}$  is the minimal eigenvalue of  $H_k$  and set  $\widehat{H} = H_k + \beta I$ .
4. Compute the search direction  $d_k = -\widehat{H}^{-1} g_k$ .
5. Perform line-search in direction  $d_k$ . Denote the step-length by  $s_k$ .
6. Set  $x_{k+1} = x_k + s_k d_k$ .

The step-length  $s$  in direction  $d$  is calculated by a gradient free line-search algorithm that tries to satisfy the Armijo condition. Obviously, for a convex  $F$ , Algorithm 26.2 is just the damped Newton method, which is known to converge under standard assumptions.

If, in the non-convex case, the Cholesky factorization in Step 2 fails, we calculate the value of  $\beta$  in Step 3 in the following way:

**Algorithm 26.3.** For a given  $\beta_0 > 0$ :

1. Set  $\beta = \beta_0$ .
2. Try to factorize  $H + \beta I$  by the Cholesky method.
3. If the factorization fails due to a negative pivot element, go to step 4, otherwise go to step 5.
4. If  $\beta \geq \beta_0$ , set  $\beta = 2\beta$  and continue with 2. Otherwise go to step 6.
5. If  $\beta \leq \beta_0$ , set  $\beta = \frac{\beta}{2}$  and continue with step 2. Otherwise STOP.
6. Set  $\beta = 2\beta$  and STOP.

Obviously, when Algorithm 26.3 terminates we have  $\beta \in [-\lambda_{\min}, -2\lambda_{\min}]$ . It is well known from the nonlinear programming literature that under quite mild assumptions any cluster point of the sequence generated by Algorithm 26.2 is a first order critical point of problem in Step 1 of Algorithm 26.1.

*Remark 26.2.* There is one exception, when we use a different strategy for the calculation of  $\beta$ . The exception is motivated by the observation that the quality of the search direction gets poor, if we choose  $\beta$  too close to  $-\lambda_{\min}$ . Therefore, if we encounter bad quality of the search direction, we use a bisection technique to calculate an approximation of  $\lambda_{\min}$ , denoted by  $\lambda_{\min}^a$ , and replace  $\beta$  by  $-1.5\lambda_{\min}^a$ .

*Remark 26.3.* Whenever we will speak about the Newton method or Newton system, later in the chapter, we will always have in mind the modified method described above.

## 26.2.6 How to Solve the Linear Systems

In both algorithms proposed in the preceding sections one has to solve repeatedly linear systems of the form

$$(H + D)d = -g, \quad (26.14)$$

where  $D$  is a diagonal matrix chosen such that the matrix  $H + D$  is positive definite. There are two categories of methods, which can be used to solve problems of type (26.14): direct and iterative methods. Let us first concentrate on the direct methods.

### 26.2.6.1 Cholesky Method

Since the system matrix in (26.14) is always positive definite, our method of choice is the Cholesky method. Depending on the sparsity structure of  $H$ , we use two different realizations:

- If the fill-in of the Hessian is below 20%, we use a sparse Cholesky solver which is based on ideas of Ng and Peyton [27]. The solver makes use of the fact that the sparsity structure is the same in each Newton step in all iterations. Hence the sparsity pattern of  $H$ , reordering of rows and columns to reduce the fill-in in the Cholesky factor, and symbolic factorization of  $H$  are all performed just once at the beginning of Algorithm 26.1. Then, each time the system (26.14) has to be solved, the numeric factorization is calculated based on the precalculated symbolic factorization. Note that we added stabilization techniques described in [34] to make the solver more robust for almost singular system matrices.
- Otherwise, if the Hessian is dense, we use the ATLAS implementation of the LAPACK Cholesky solver DPOTRF.

### 26.2.6.2 Iterative Methods

We solve the system  $\widehat{H}d = -g$  with a symmetric positive definite and, possibly, ill-conditioned matrix  $\widehat{H} = H + D$ . We use the very standard preconditioned conjugate

gradient method. The algorithm is well known and we will not repeat it here. The algorithm is stopped when the normalized residuum is sufficiently small:

$$\|\widehat{H}d_k + g\|/\|g\| \leq \epsilon.$$

In our tests, the choice  $\epsilon = 5 \cdot 10^{-2}$  was sufficient.

### 26.2.6.3 Preconditioners

We are looking for a preconditioner – a matrix  $M \in \mathbb{S}_+^n$  – such that the system  $M^{-1}\widehat{H}d = -M^{-1}g$  can be solved more efficiently than the original system  $\widehat{H}d = -g$ . Apart from standard requirements that the preconditioner should be efficient and inexpensive, we also require that it should only use Hessian-vector products. This is particularly important in the case when we want to use the Hessian-free version of the algorithm.

#### Diagonal Preconditioner

This is a simple and often-used preconditioner with

$$M = \text{diag}(\widehat{H}).$$

On the other hand, being simple and general, it is not considered to be very efficient. Furthermore, we need to know the diagonal elements of the Hessian. It is certainly possible to compute these elements by Hessian-vector products. For that, however, we would need  $n$  gradient evaluations and the approach would become too costly.

#### L-BFGS Preconditioner

Introduced by Morales–Nocedal [25], this preconditioner is intended for application within the Newton method. (In a slightly different context, the (L-)BFGS preconditioner was also proposed in [10].) The algorithm is based on the limited-memory BFGS formula ([28]) applied to successive CG (instead of Newton) iterations. The preconditioner, as used in PENNON is described in detail in [18]. Here we only point out some important features.

As recommended in the standard L-BFGS method, we used 16–32 correction pairs, if they were available. Often the CG method finished in less iterations and in that case we could only use the available iterations for the correction pairs. If the number of CG iterations is higher than the required number of correction pairs  $\mu$ , we may ask how to select these pairs. We have two options: Either we take the last  $\mu$  pairs or an “equidistant” distribution over all CG iterations. The second option is slightly more complicated but we may expect it to deliver better results.

The L-BFGS preconditioner has the big advantage that it only needs Hessian-vector products and can thus be used in the Hessian-free approaches.

On the other hand, it is more complex than the above preconditioners; also our results are not conclusive concerning the efficiency of this approach. For many problems it worked satisfactorily, for some, on the other hand, it even lead to higher number of CG steps than without preconditioner.

### 26.2.7 Multiplier and Penalty Update

For the penalty function  $\Phi_p$  from (26.10), the formula for update of the matrix multiplier  $U$  in Step (ii) of Algorithm 26.1 reduces to

$$U^{k+1} = (p^k)^2 \mathcal{Z}(x^{k+1}) U^k \mathcal{Z}(x^{k+1}) \quad (26.15)$$

with  $\mathcal{Z}$  defined as in (26.11). Note that when  $U^k$  is positive definite, so is  $U^{k+1}$ . We set  $U^1$  equal to a positive multiple of the identity.

Numerical tests indicate that big changes in the multipliers should be avoided for the following reasons. Big change of  $U$  means big change of the augmented Lagrangian that may lead to a large number of Newton steps in the subsequent iteration. It may also happen that already after few initial steps the multipliers become ill-conditioned and the algorithm suffers from numerical difficulties. To overcome these, we do the following:

1. Calculate  $U^{k+1}$  using (26.15).
2. Choose a positive  $\mu_A \leq 1$ , typically 0.5.
3. Compute  $\lambda_A = \min\left(\mu_A, \mu_A \frac{\|U^k\|_F}{\|U^{k+1} - U^k\|_F}\right)$ .
4. Update the current multiplier by

$$U^{new} = U^k + \lambda_A(U^{k+1} - U^k).$$

Given an initial iterate  $x^1$ , the initial penalty parameter  $p^1$  is chosen large enough to satisfy the inequality

$$p^1 I - \mathcal{A}(x^1) > 0.$$

Let  $\lambda_{\max}(\mathcal{A}(x^{k+1})) \in (-\infty, p^k)$  denote the maximal eigenvalue of  $\mathcal{A}(x^{k+1})$ ,  $\pi < 1$  be a constant factor, depending on the initial penalty parameter  $p^1$  (typically chosen between 0.3 and 0.6) and  $x_{\text{feas}}$  be a feasible point. Let  $l$  be set to 0 at the beginning of Algorithm 26.1. Using these quantities, our strategy for the penalty parameter update can be described as follows:

1. If  $p^k < p_{eps}$ , set  $\gamma = 1$  and go to 6.
2. Calculate  $\lambda_{\max}(\mathcal{A}(x^{k+1}))$ .
3. If  $\pi p^k > \lambda_{\max}(\mathcal{A}(x^{k+1}))$ , set  $\gamma = \pi$ ,  $l = 0$  and go to 6.

4. If  $l < 3$ , set  $\gamma = (\lambda_{\max}(\mathcal{A}(x^{k+1})) + p^k) / (2p^k)$ , set  $l := l + 1$  and go to 6.
5. Let  $\gamma = \pi$ , find  $\lambda \in (0, 1)$  such, that

$$\lambda_{\max}(\mathcal{A}(\lambda x^{k+1} + (1 - \lambda)x_{\text{feas}})) < \pi p^k,$$

set  $x^{k+1} = \lambda x^{k+1} + (1 - \lambda)x_{\text{feas}}$  and  $l := 0$ .

6. Update current penalty parameter by  $p^{k+1} = \gamma p^k$ .

The reasoning behind steps 3 to 5 is as follows: As long as the inequality

$$\lambda_{\max}(\mathcal{A}(x^{k+1})) < \pi p^k \quad (26.16)$$

holds, the values of the augmented Lagrangian in the next iteration remain finite and we can reduce the penalty parameter by the predefined factor  $\pi$ . As soon as inequality (26.16) is violated, an update using  $\pi$  would result in an infinite value of the augmented Lagrangian in the next iteration. Therefore the new penalty parameter should be chosen from the interval  $(\lambda_{\max}(\mathcal{A}(x^{k+1})), p^k)$ . Because a choice close to the left boundary of the interval leads to large values of the augmented Lagrangian, while a choice close to the right boundary slows down the algorithm, we choose  $\gamma$  such that

$$p^{k+1} = \frac{\lambda_{\max}(\mathcal{A}(x^{k+1})) + p^k}{2}.$$

In order to avoid stagnation of the penalty parameter update process due to repeated evaluations of step 4, we redefine  $x^{k+1}$  using the feasible point  $x_{\text{feas}}$  whenever step 4 is executed in three successive iterations; this is controlled by the parameter  $l$ . If no feasible point is yet available, Algorithm 26.1 is stopped and restarted from the scratch with a different choice of initial multipliers. The parameter  $p_{eps}$  is typically chosen as  $10^{-6}$ . In case we detect problems with convergence of Algorithm 26.1,  $p_{eps}$  is decreased and the penalty parameter is updated again, until the new lower bound is reached.

## 26.2.8 Initialization and Stopping Criteria

### 26.2.8.1 Initialization

Algorithm 26.1 can start with an arbitrary primal variable  $x \in \mathbb{R}^n$ . Therefore we simply choose  $x^1 = 0$ . For the description of the multiplier initialization strategy we rewrite problem (SDP) in the following form:

$$\begin{aligned} & \min_{x \in \mathbb{R}^n} f(x) \\ & \text{subject to} \\ & \mathcal{A}_i(x) \leq 0, \quad i = 1, \dots, \sigma. \end{aligned} \quad (26.17)$$

Here  $\mathcal{A}_i(x) \in \mathbb{S}^{m_j}$  are diagonal blocks of the original constrained matrix  $\mathcal{A}(x)$  and we have  $\sigma = 1$  if  $\mathcal{A}(x)$  consists of only one block. Now the initial values of the multipliers are set to

$$U_j^1 = \mu_j I_{m_j}, \quad j = 1, \dots, \sigma,$$

where  $I_{m_j}$  are identity matrices of order  $m_j$  and

$$\mu_j = m_j \max_{1 \leq \ell \leq n} \frac{1 + \left| \frac{\partial f(x)}{\partial x_\ell} \right|}{1 + \left\| \frac{\partial \mathcal{A}(x)}{\partial x_\ell} \right\|}. \quad (26.18)$$

Given the initial iterate  $x^1$ , the initial penalty parameter  $p^1$  is chosen large enough to satisfy the inequality

$$p^1 I - \mathcal{A}(x^1) > 0.$$

### 26.2.8.2 Stopping Criterion in the Sub-problem

In the first iterations of Algorithm 26.1, the approximate minimization of  $F$  is stopped when  $\|\frac{\partial}{\partial x} F(x, U, p)\| \leq \alpha$ , where  $\alpha = 0.01$  is a good choice in most cases. In the remaining iterations, after a certain precision is reached,  $\alpha$  is reduced in each outer iteration by a constant factor, until a certain  $\underline{\alpha}$  (typically  $10^{-7}$ ) is reached.

### 26.2.8.3 Stopping Criterion for the Main Algorithm

We have implemented two different stopping criteria for the main algorithm.

- *First alternative:* The main algorithm is stopped if both of the following inequalities hold:

$$\frac{|f(x^k) - F(x^k, U^k, p)|}{1 + |f(x^k)|} < \varepsilon_1, \quad \frac{|f(x^k) - f(x^{k-1})|}{1 + |f(x^k)|} < \varepsilon_1,$$

where  $\varepsilon_1$  is typically  $10^{-7}$ .

- *Second alternative:* The second stopping criterion is based on the KKT-conditions. Here the algorithm is stopped, if

$$\min \{ \lambda_{\max}(\mathcal{A}(x)), |\langle \mathcal{A}(x), U \rangle|, \|\nabla_x F(x, U, p)\| \} \leq \varepsilon_2.$$

## 26.2.9 Complexity

The computational complexity of Algorithm 26.1 is clearly dominated by Step 1. In each step of the Newton method, there are two critical issues: assembling of the Hessian of the augmented Lagrangian and solution of the linear system of equations (the Newton system).

### 26.2.9.1 Hessian Assembling

Full Matrices

Assume first that all the data matrices are full. The assembling of the Hessian (26.13) can be divided into the following steps:

- Calculation of  $\mathcal{Z}(x) \rightarrow O(m^3 + m^2n)$ .
- Calculation of  $\mathcal{Z}(x)U\mathcal{Z}(x) \rightarrow O(m^3)$ .
- Calculation of  $\mathcal{Z}(x)U\mathcal{Z}(x)\mathcal{A}_i'(x)\mathcal{Z}(x)$  for all  $i \rightarrow O(m^3n)$ .
- Assembling the rest  $\rightarrow O(m^2n^2)$ .

Now it is straightforward to see that an estimate of the complexity of assembling of (26.13) is given by  $O(m^3n + m^2n^2)$ .

Many optimization problems, however, have very sparse data structure and therefore have to be treated by sparse linear algebra routines. We distinguish three basic types of sparsity.

#### The Block Diagonal Case

The first case under consideration is the block diagonal case. In particular, we want to describe the case, where:

- The matrix  $\mathcal{A}(x)$  consists of many (small) blocks.

In this situation the original SDP problem (26.1) can be written in the form (26.17). If we define  $\bar{m} = \max\{m_i \mid i = 1, \dots, d\}$  we can estimate the computational complexity of the Hessian assembling by  $O(d\bar{m}^3n + d\bar{m}^2n^2)$ . An interesting subcase of problem (26.17) is when:

- Each of the matrix constraints  $\mathcal{A}_i(x)$  involves just a few components of  $x$ .

If we denote the maximal number of components of  $x$  on which each of the blocks  $\mathcal{A}_i(x), i = 1, 2, \dots, d$  depends by  $\bar{n}$ , our complexity formula becomes  $O(d\bar{m}^3\bar{n} + d\bar{m}^2\bar{n}^2)$ . If we further assume that the numbers  $\bar{n}$  and  $\bar{m}$  are of order  $O(1)$ , then the complexity estimate can be further simplified to  $O(d)$ . A typical example for this sparsity class are the “mater” problems discussed in Sect. 26.3.

### The Case When $\mathcal{A}(x)$ Is Dense and $\mathcal{A}'_i(x)$ are Sparse

Let us first mention that for any index pair  $(i, j) \in \{1, \dots, n\} \times \{1, \dots, n\}$  the non-zero structure of the matrix  $\mathcal{A}_{i,j}''(x)$  is given by (a subset of the) intersection of the non-zero index sets of the matrices  $\mathcal{A}'_i(x)$  and  $\mathcal{A}'_j(x)$ . We assume that:

- There are at most  $O(1)$  non-zero entries in  $\mathcal{A}'_i(x)$  for all  $i = 1, \dots, n$ .

Then the calculation of the term

$$\left[ \langle \mathcal{Z}(x)U\mathcal{Z}(x), \mathcal{A}_{i,j}''(x) \rangle \right]_{i,j=1}^n$$

can be performed in  $O(n^2)$  time. In the paper by Fujisawa, Kojima and Nakata on exploiting sparsity in semidefinite programming [8] several ways are presented how to calculate a matrix of the form

$$D_1 S_1 D_2 S_2 \tag{26.19}$$

efficiently, if  $D_1$  and  $D_2$  are dense and  $S_1$  and  $S_2$  are sparse matrices. If our assumption above holds, the calculation of the matrix

$$\left[ \langle \mathcal{Z}(x)U\mathcal{Z}(x)\mathcal{A}'_j(x)\mathcal{Z}(x), \mathcal{A}'_i(x) \rangle \right]_{i,j=1}^n$$

can be performed in  $O(n^2)$  time. Thus, recalling that for the calculation of  $\mathcal{Z}(x)$  we have to compute the inverse of an  $(m \times m)$ -matrix, we get the following complexity estimate for the Hessian assembling:  $O(m^3 + n^2)$ . Note that in our implementation we follow the ideas presented in [8]. Many linear SDP problems coming from real world applications have exactly the sparsity structure discussed in this paragraph.

### The Case When $\mathcal{A}(x)$ and the Cholesky Factor of $\mathcal{A}(x)$ Is Sparse

Also in this case we can conclude that all partial derivatives of  $\mathcal{A}(x)$  of first and second order are sparse matrices. Therefore it suffices to assume that:

- The matrix  $\mathcal{A}(x)$  has at most  $O(1)$  non-zero entries.

We have to compute expressions of type

$$(\mathcal{A}(x) - pI)^{-1} U (\mathcal{A}(x) - pI)^{-1} \quad \text{and} \quad (\mathcal{A}(x) - pI)^{-1}.$$

Note that each of the matrices above can be calculated by maximally two operations of the type  $(A - I)^{-1}M$ , where  $M$  is a symmetric matrix. Now assume that not only  $\mathcal{A}(x)$  but also its Cholesky factor is sparse. Then, obviously, the Cholesky

factor of  $(\mathcal{A}(x) - pI)$ , denoted by  $L$ , will also be sparse. This leads to the following assumption:

- Each column of  $L$  has at most  $O(1)$  non-zero entries.

Now the  $i$ -th column of  $C := (\mathcal{A}(x) - pI)^{-1}M$  can then be computed as

$$C^i = (L^{-1})^T L^{-1} M^i, \quad i = 1, \dots, n,$$

and the complexity of computing  $C$  by Cholesky factorization is  $O(m^2)$ , compared to  $O(m^3)$  when computing the inverse of  $(A(x) - pI)$  and its multiplication by  $U$ . So the overall complexity of Hessian assembling is of order  $O(m^2 + n^2)$ .

*Remark 26.4.* Recall that in certain cases, we do not need to assemble the Hessian matrix. In this case the complexity estimates can be improved significantly; see the next section.

### 26.2.9.2 Solution of the Newton System

As mentioned above, the Newton system

$$Hd = -g \tag{26.20}$$

can either be solved by a direct (Cholesky) solver or by an iterative method.

#### Cholesky Method

The complexity of Cholesky algorithm is  $O(n^3)$  for dense matrices and  $O(n^\kappa)$ ,  $1 \leq \kappa \leq 3$  for sparse matrices, where  $\kappa$  depends on the sparsity structure of the matrix, going from a diagonal to a full matrix.

#### Iterative Algorithms

From the complexity viewpoint, the only demanding step in the CG method is a matrix-vector product with a matrix of dimension  $n$ . For a dense matrix and vector, it needs  $O(n^2)$  operations. Theoretically, in exact arithmetics, the CG method needs  $n$  iterations to find an exact solution of the system, hence it is equally expensive as the Cholesky algorithm. There are, however, two points that may favor the CG method.

First, it is well known that the convergence behavior of the CG method can be significantly improved by preconditioning. The choice of the preconditioner  $M$  will be the subject of the next section.

The second – and very important – point is that we actually do not need an exact solution of the Newton system. On the contrary, a rough approximation of it will do (see [11, Theorem 10.2]). Hence, in practice, we may need just a few CG iterations to reach the required accuracy. This is in contrast with the Cholesky method where we cannot control the accuracy of the solution and always have to compute the exact one (within the machine precision). Note that we always start the CG method with initial approximation  $d_0 = 0$ ; thus, performing just one CG step, we would obtain the steepest descend method. Doing more steps, we improve the search direction toward the Newton direction; note the similarity to the Toint–Steihaug method [28].

Summarizing these two points: when using the CG algorithm, we may expect to need just  $O(n^2)$  operations, at least for well-conditioned (or well-preconditioned) systems.

Note that we are still talking about dense problems. The use of the CG method is a bit nonstandard in this context – usually it is preferable for large sparse problems. However, due to the fact that we just need a very rough approximation of the solution, we may favor it to the Cholesky method also for medium-sized dense problems.

### Approximate Hessian Formula

When solving the Newton system by the CG method, the Hessian is only needed in a matrix-vector product of the type  $Hv := \nabla^2 F(x^k)v$ . Because we only need to compute the products, we may use a finite difference formula for the approximation of this product

$$\nabla^2 F(x^k)v \approx \frac{\nabla F(x^k + hv) - \nabla F(x^k)}{h} \quad (26.21)$$

with  $h = (1 + \|x^k\|_2 \sqrt{\varepsilon})$ ; see [28]. In general,  $\varepsilon$  is chosen so that the formula is as accurate as possible and still not influenced by round-off errors. The “best” choice is obviously case dependent; in our implementation, we use  $\varepsilon = 10^{-6}$ . Hence the complexity of the CG method amounts to the number of CG iterations times the complexity of gradient evaluation, which is of order  $O(m^3 + Kn)$ , where  $K$  denotes the maximal number of nonzero entries in  $\mathcal{A}'_i(x), i = 1, 2, \dots, n$ . This may be in sharp contrast with the Cholesky method approach when we have to compute the full Hessian *and* solve the system by Cholesky method. Again, we have the advantage that we do not have to store the Hessian in the memory.

This approach is clearly not always applicable. With certain SDP problems it may happen that the Hessian computation is not much more expensive than the gradient evaluation. In this case the Hessian-free approach may be rather time-consuming. Indeed, when the problem is ill-conditioned and we need many CG iterations, we have to evaluate the gradient many (thousand) times. On the other hand, when using Cholesky method, we compute the Hessian just once.

## 26.3 PENSDP

When both functions in (26.1) are linear, the problem simplifies to a standard (primal or dual, as you like) linear semidefinite programming problem (LSDP)

$$\min_{x \in \mathbb{R}^n} f^T x$$

subject to

$$\sum_{k=1}^n x_k A_k - A_0 \leq 0. \quad (26.22)$$

Here we write explicitly all matrix inequality constraints, as well as linear constraints, in order to introduce necessary notation. In the next sections, we will present some special features of the code PENSDP designed to solve (26.22), as well as selected numerical examples demonstrating its capabilities.

### 26.3.1 The Code PENSDP

#### 26.3.1.1 Special Features

##### Stopping Criteria

In the case of linear semidefinite programs, we have additionally adopted the DIMACS criteria [24]. To define these criteria, we denote  $\tilde{\mathcal{A}}(x) = \sum_{k=1}^n x_k A_k$ . Recall that  $U$  is the corresponding Lagrangian multiplier and let  $\tilde{\mathcal{A}}^*(\cdot)$  denote the adjoint operator to  $\tilde{\mathcal{A}}(\cdot)$ . The DIMACS error measures are defined as

$$\text{err}_1 = \frac{\|\tilde{\mathcal{A}}^*(U) - f\|}{1 + \|f\|}$$

$$\text{err}_2 = \max \left\{ 0, \frac{-\lambda_{\min}(U)}{1 + \|f\|} \right\} \quad \text{err}_4 = \max \left\{ 0, \frac{-\lambda_{\min}(\tilde{\mathcal{A}}(x) - A_0)}{1 + \|A_0\|} \right\}$$

$$\text{err}_5 = \frac{\langle A_0, U \rangle - f^T x}{1 + |\langle A_0, U \rangle| + |f^T x|} \quad \text{err}_6 = \frac{\langle \tilde{\mathcal{A}}(x) - A_0, U \rangle}{1 + |\langle A_0, U \rangle| + |f^T x|}.$$

Here,  $\text{err}_1$  represents the (scaled) norm of the gradient of the Lagrangian,  $\text{err}_2$  and  $\text{err}_4$  is the dual and primal infeasibility, respectively, and  $\text{err}_5$  and  $\text{err}_6$  measure the duality gap and the complementarity slackness. Note that, in our code,  $\text{err}_2 = 0$  by definition; also  $\text{err}_3$  that involves the slack variable (not used in our problem

formulation) is automatically zero. If the “DIMACS stopping criterion” is activated we require that

$$\text{err}_k \leq \delta_{\text{DIMACS}}, \quad k \in \{1, 4, 5, 6\}.$$

### Implicit Hessian Formula

As mentioned before, when solving the Newton system by the CG method, the Hessian is only needed in a matrix-vector product of the type  $Hv := \nabla^2 F(x^k)v$ . Instead of computing the Hessian matrix explicitly and then multiplying it by a vector  $v$ , we can use the following formula for the Hessian-vector multiplication

$$\nabla^2 F(x^k)v = 2\mathcal{A}^* \left( (p^k)^2 \mathcal{Z}(x^k) U^k \mathcal{Z}(x^k) \mathcal{A}(v) \mathcal{Z}(x^k) \right), \quad (26.23)$$

where we assume that  $\mathcal{A}$  is linear of the form  $A(x) = \sum_{i=1}^n x_i A_i$  and  $\mathcal{A}^*$  denotes its adjoint. Hence, in each CG step, we only have to evaluate matrices  $\mathcal{A}(v)$  (which is simple),  $\mathcal{Z}(x^k)$  and  $\mathcal{Z}(x^k)U^k\mathcal{Z}(x^k)$  (which are needed in the gradient computation, anyway), and perform two additional matrix-matrix products. The resulting complexity formula for one Hessian-vector product is thus  $O(m^3 + Kn)$ , where again  $K$  denotes the maximal number of nonzero entries in  $\mathcal{A}'_i(x)$ ,  $i = 1, 2, \dots, n$ .

The additional (perhaps the main) advantage of this approach is the fact that we do not have to store the Hessian in the memory, thus the memory requirements (often the real bottleneck of SDP codes) are drastically reduced.

### Dense Versus Sparse

For the efficiency of PENSDP, it is important to know if the problem has a sparse or dense Hessian. The program can check this automatically. The check, however, may take some time and memory, so if the user knows that the Hessian is dense (and this is the case of most problems), this check can be avoided. This, for certain problems, can lead to substantial savings not only in CPU time but also in memory requirements.

### Hybrid Mode

For linear semidefinite programming problems, we use the following hybrid approach, whenever the number of variables  $n$  is large compared to the size of the matrix constraint  $m$ : We try to solve the linear systems using the iterative approach as long as the iterative solver needs a moderate number of iterations. In our current implementation the maximal number of CG iterations allowed is 100. Each time the maximal number of steps is reached, we solve the system again by the Cholesky

method. Once the system is solved by the Cholesky method, we use the Cholesky factor as a preconditioner for the iterative solver in the next system. As soon as the iterative solver fails three times in sequel, we completely switch to the Cholesky method.

The hybrid mode allows us to reach a high precision solution while keeping the solution time low. The main reason is that, when using the iterative approach, the Hessian of the Augmented Lagrangian has not to be calculated explicitly.

### 26.3.1.2 User Interfaces

The user has a choice of several interfaces to PENSDP.

#### SDPA Interface

The problem data are written in an ASCII input file in a SDPA sparse format, as introduced in [9]. The code needs an additional ASCII input file with parameter values.

#### C/C++/FORTRAN Interface

PENSDP can also be called as a function (or subroutine) from a C, C++ or FORTRAN program. In this case, the user should link the PENSDP library to his/her program. In the program the user then has to specify problem dimensions, code parameters and the problem data (vectors and matrices) in a sparse format.

#### MATLAB Interface

In MATLAB, PENSDP is called with the following arguments:

```
[f,x,u,iflag,niter,feas] = pensdpm(pen);
```

where pen a MATLAB structure array with fields describing the problem dimensions and problem data, again in a sparse format.

#### YALMIP Interface

The most comfortable way of preparing the data and calling PENSDP is via YALMIP [22]. YALMIP is a modeling language for advanced modeling and solution of convex and nonconvex optimization problems. It is implemented as a free toolbox for MATLAB. When calling PENSDP from YALMIP, the user does not have to bother

with the sparsity pattern of the problem – any linear optimization problem with vector or matrix variables will be translated by YALMIP into PENSDP data structure.

### 26.3.2 Numerical Experiments

It is not our goal to compare PENSDP with other linear SDP solvers. This is done elsewhere in this book and the reader can also consult the benchmark page of Hans Mittelmann,<sup>1</sup> containing contemporary results. We will thus present only results for selected problems and will concentrate on the effect of special features available in PENSDP. The results for the “mater” and “rose13” problems were obtained on an Intel Core i7 processor 2.67 GHz with 4 GB memory.

#### 26.3.2.1 Sparsity: “mater” Problems

Let us consider the “mater\*” problems from Mittelmann’s collection.<sup>2</sup> These problems are significant by several different sparsity patterns of the problem data. The problem has many small matrix constraints, the data matrices are sparse, only very few variables are involved in each constraint and the resulting Hessian matrix is sparse. We cannot switch off sparsity handling in routines for Hessian assembling but we can run the code with (forced use of) dense Cholesky factorization and with sparse Cholesky routine. For instance, problem “mater3” with 1,439 variables and 328 matrix constraints of size 11 was solved in 32 s using the dense Cholesky and only 4 s using the sparse Cholesky routine. The difference is, of course, more dramatic for larger problems. The next problem “mater4” has 4,807 variables and 1,138 matrix constraints of size 11. While the sparse version of PENSDP only needed 20 s to solve it, the dense version needed 1,149 s. And while the largest problem “mater6” (20,463 variables and 4,968 matrix constraints) does not even fit in the 4 GB memory for the dense version, the sparse code needs only 100 MB and solves the problem in 134 s.

#### 26.3.2.2 Iterative Solver: “TOH” Collection

The effect of the use of preconditioned conjugate gradient method for the solution of the Newton system was described in detail in [18, 19]. Recall that iterative solvers are suitable for problems with a large number of variable and relatively small constraint matrices. We select from [18, 19] two examples arising from maximum clique problems on randomly generated graphs (the “TOH” collection in [18]).

---

<sup>1</sup>[plato.la.asu.edu/bench.html](http://plato.la.asu.edu/bench.html).

<sup>2</sup>[plato.asu.edu/ftp/sparse\\_sdp.html](http://plato.asu.edu/ftp/sparse_sdp.html).

The first example is “theta62” with 13,390 variables and matrix size 300. This problem could still be solved using the direct (dense Cholesky) solver and the code needed 13,714 s to solve it. Compared to that, the iterative version of the code only needed 40 s to obtain the solution with the same precision. The average number of CG steps in each Newton system was only 10. The largest problem solved in the paper was “theta162” with 127,600 variables and a matrix constraint of size 800. Note that the Hessians of this example is *dense*, so to solve the problem by the direct version of PENSDP (or by any other interior-point algorithm) would mean to store and factorize a full matrix of dimension 127,600 by 127,600. On the other hand, the iterative version of PENSDP, being effectively a first-order code, has only modest memory requirements and allowed us to solve this problem in only 672 s.

### 26.3.2.3 Hybrid Mode: “rose13”

To illustrate the advantages of the hybrid mode, we consider the problem “rose13” from Mittelmann’s collection.<sup>3</sup> The problem has 2,379 variables and one matrix constraint of size 105. When we solve the problem by PENSDP with a direct solver of the Newton system, the code needs 17 global iterations, 112 Newton steps and the solution is obtained in 188 s CPU time, 152 s of which is spent in the Cholesky factorization routine.

Let us now solve the problem using the iterative solver for the Newton systems. Below we see the first and the last iterations of PENSDP. The required precision of DIMACS criteria is  $\delta_{\text{DIMACS}} = 10^{-3}$ .

*	it	obj	opt	Nwt	CG	*
	0	0.0000e+000	0.0000e+000	0	0	
	1	1.8893e+003	8.3896e+000	10	321	
	2	2.2529e+002	8.2785e+000	17	1,244	
...						
	9	-1.1941e+001	2.2966e+000	36	9,712	
	10	-1.1952e+001	4.9578e+000	46	10,209	
...						
	15	-1.1999e+001	5.0429e-002	119	103,905	
	16	-1.1999e+001	4.4050e-003	134	167,186	

The table shows the global iterations of Algorithm 26.1, the value of the objective function and the gradient of the augmented Lagrangian and, in the last two columns,

---

<sup>3</sup>[plato.asu.edu/ftp/sparse\\_sdp.html](http://plato.asu.edu/ftp/sparse_sdp.html).

the cumulative number of Newton and CG steps. The code needed a large number of CG steps that was growing with increasing conditioning of the Newton system. The problem was solved in 732 s of CPU time. When we try to solve the same problem with a higher precision ( $\delta_{\text{DIMACS}} = 10^{-7}$ ), the iterative method, and consequently the whole algorithm, will get into increasing difficulties. Below we see the last two iterations of PENSDP before it was stopped due to one-hour time limit.

```
...
| 28| -1.2000e+001 | 5.2644e-002 | 373 | 700,549 |
| 29| -1.2000e+001 | 3.0833e-003 | 398 | 811,921 |
*****
```

We can see that the optimality criterium is actually oscillating around  $10^{-3}$ .

We now switch the hybrid mode on. The difference will be seen already in the early iterations of PENSDP. Running the problem with  $\delta_{\text{DIMACS}} = 10^{-3}$ , we get the following output

```
*****
* it |     obj      |     opt      |   Nwt |   CG   *
*****
|  0|  0.0000e+000 | 0.0000e+000 |    0 |    0 |
|  1|  1.8893e+003 | 8.3896e+000 |   10 |  321 |
|  2|  2.3285e+002 | 1.9814e+000 |   18 |  848 |
...
|  9| -1.1971e+001 | 4.5469e-001 |   36 | 1,660 |
| 10| -1.1931e+001 | 7.4920e-002 |   63 | 2,940 |
...
| 13| -1.1998e+001 | 4.1400e-005 |  104 | 5,073 |
| 14| -1.1999e+001 | 5.9165e-004 |  115 | 5,518 |
*****
```

The CPU time needed was only 157 s, 130 of which were spent in the Cholesky factorization routine. When we now increase the precision to  $\delta_{\text{DIMACS}} = 10^{-7}$ , the PENSDP with the hybrid mode will only need a few more iterations to reach it:

```
...
| 16| -1.2000e+001 | 9.8736e-009 | 142 | 6,623 |
| 17| -1.2000e+001 | 5.9130e-007 | 156 | 7,294 |
*****
```

The total CPU time increased to 201 s, 176 of which were spent in Cholesky factorization.

Notice that the difference between the direct solver and the hybrid method would be even more significant for larger problems, such as “rose15”.

## 26.4 PENBMI

We solve the SDP problem with quadratic objective function and linear and bilinear matrix inequality constraints:

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} x^T Q x + f^T x$$

subject to

$$\sum_{k=1}^n b_k^i x_k \leq c^i, \quad i = 1, \dots, N_\ell$$

$$A_0^i + \sum_{k=1}^n x_k A_k^i + \sum_{k=1}^n \sum_{\ell=1}^n x_k x_\ell K_{k\ell}^i \leq 0, \quad i = 1, \dots, N, \quad (26.24)$$

where all data matrices are from  $\mathbb{S}^m$ .

### 26.4.1 The Code PENBMI

#### 26.4.1.1 User Interface

The advantage of our formulation of the BMI problem is that, although nonlinear, the data only consist of matrices and vectors, just like in the linear SDP case. The user does not have to provide the (first and second) derivatives of the matrix functions, as these are readily available. Hence the user interface of PENBMI is a direct extension of the interface to PENSMP described in the previous section.

In particular, the user has the choice of calling PENBMI from Matlab or from a C/C++/Fortran code. In both cases, the user has to specify the matrices  $A_k^i, k = 1, \dots, n$ , and  $K_{k\ell}^i, k, \ell = 1, \dots, n$ , for all constraints  $i = 1, \dots, N$ , matrix  $Q$  from the objective function and vectors  $f, c, b^i, i = 1, \dots, N$ . As in the linear SDP case, all matrices and vectors are assumed to be sparse (or even void), so the user has to provide the sparsity pattern of the constraints (which matrices are present) and sparsity structure of each matrix.

Again, the most comfortable way of preparing the data and calling PENBMI is via YALMIP. In this case, the user does not have to stick to the formulation (26.24) and bother with the sparsity pattern of the problem – any optimization problem with vector or matrix variables and linear or quadratic objective function and (matrix) constraints will be translated by YALMIP into formulation (26.24) and the corresponding user interface will be automatically created. Below is a simple example of YALMIP code for the LQ optimal feedback problem formulated as

$$\begin{aligned} & \min_{P \in \mathbb{R}^{2 \times 2}, K \in \mathbb{R}^{1 \times 2}} \text{trace}(P) \\ \text{s.t.} \quad & (A + BK)^T P + P(A + BK) < -I_{2 \times 2} - K^T K \\ & P > 0 \end{aligned}$$

with

$$A = \begin{pmatrix} -1 & 2 \\ -3 & -4 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Using YALMIP, the problem is formulated and solved by the following few lines

```
>> A = [-1 2;-3 -4]; B = [1;1];
>> P = sdpvar(2,2); K = sdpvar(1,2);
>> F = set(P>0)+set((A+B*K)'*P+P*(A+B*K) < -eye(2)-K'*K);
>> solvesdp(F, trace(P), sdpsettings('solver','penbmi'));
```

### 26.4.2 The Static Output Feedback Problem

Many interesting problems in linear and nonlinear systems control cannot be formulated and solved as LSDP. BMI formulation of the control problems was made popular in the mid 1990s [12]; there were, however, no computational methods for solving non-convex BMIs, in contrast with convex LMIs for which powerful interior-point algorithms were available.

The most fundamental of these problems is perhaps static output feedback (SOF) stabilization: given a triplet of matrices  $A, B, C$  of suitable dimensions, find a matrix  $F$  such that the eigenvalues of matrix  $A + BFC$  are all in a given region of the complex plane, say the open left half-plane [3].

No LSDP formulation is known for this problem but a straightforward application of Lyapunov's stability theory leads to a BMI formulation: matrix  $A + BFC$  has all its eigenvalues in the open left half-plane if and only if there exists a matrix  $X$  such that

$$(A + BFC)^T X + (A + BFC)X < 0, \quad X = X^T > 0$$

where  $< 0$  and  $> 0$  stand for positive and negative definite, respectively.

We present a short description of the benchmark collection *COMPlib*: the COnstrained Matrix–optimization Problem library [20].<sup>4</sup> *COMPlib* can be used

<sup>4</sup>See [http://www.mathematik.uni-trier.de/~leibfritz/Proj\\_TestSet/NSDPTestSet.htm](http://www.mathematik.uni-trier.de/~leibfritz/Proj_TestSet/NSDPTestSet.htm).

as a benchmark collection for a very wide variety of algorithms solving matrix optimization problems. Currently *COMPlib* consists of 124 examples collected from the engineering literature and real-life applications for LTI control systems of the form

$$\begin{aligned}\dot{x}(t) &= Ax(t) + B_1w(t) + Bu(t), \\ z(t) &= C_1x(t) + D_{11}w(t) + D_{12}u(t), \\ y(t) &= Cx(t) + D_{21}w(t),\end{aligned}\tag{26.25}$$

where  $x \in \mathbb{R}^{n_x}$ ,  $u \in \mathbb{R}^{n_u}$ ,  $y \in \mathbb{R}^{n_y}$ ,  $z \in \mathbb{R}^{n_z}$ ,  $w \in \mathbb{R}^{n_w}$  denote the state, control input, measured output, regulated output, and noise input, respectively.

The heart of *COMPlib* is the MATLAB function file *COMPlib.m*. This function returns the data matrices  $A$ ,  $B_1$ ,  $B$ ,  $C_1$ ,  $C$ ,  $D_{11}$ ,  $D_{12}$  and  $D_{21}$  of (26.25) of each individual *COMPlib* example. Depending on specific control design goals, it is possible to derive particular matrix optimization problems using the data matrices provided by *COMPlib*. A non exhaustive list of matrix optimization problems arising in feedback control design are stated in [20]. Many more control problems leading to NSDPs, BMIs or SDPs can be found in the literature.

Here we state the BMI formulation of two basic static output feedback control design problems: SOF– $\mathcal{H}_2$  and SOF– $\mathcal{H}_\infty$ . The goal is to determine the matrix  $F \in \mathbb{R}^{n_u \times n_y}$  of the SOF control law  $u(t) = Fy(t)$  such that the closed loop system

$$\begin{aligned}\dot{x}(t) &= A(F)x(t) + B(F)w(t), \\ z(t) &= C(F)x(t) + D(F)w(t),\end{aligned}\tag{26.26}$$

fulfills some specific control design requirements, where  $A(F) = A + BFC$ ,  $B(F) = B_1 + BFD_{21}$ ,  $C(F) = C_1 + D_{12}FC$ ,  $D(F) = D_{11} + D_{12}FD_{21}$ .

We begin with the SOF– $\mathcal{H}_2$  problem: *Suppose that  $D_{11} = 0$  and  $D_{21} = 0$ . Find a SOF gain  $F$  such that  $A(F)$  is Hurwitz and the  $\mathcal{H}_2$ -norm of (26.26) is minimal.* This problem can be rewritten to the following  $\mathcal{H}_2$ -BMI problem formulation, see, e.g. [20]:

$$\begin{aligned}\min \quad & \text{Tr}(X) \quad \text{s.t.} \quad Q > 0, \\ & (A + BFC)Q + Q(A + BFC)^T + B_1B_1^T \leq 0, \\ & \begin{bmatrix} X & (C_1 + D_{12}FC)Q \\ Q(C_1 + D_{12}FC)^T & Q \end{bmatrix} \geq 0,\end{aligned}\tag{26.27}$$

where  $Q \in \mathbb{R}^{n_x \times n_x}$ ,  $X \in \mathbb{R}^{n_z \times n_z}$ .

$\mathcal{H}_\infty$  synthesis is an attractive model-based control design tool and it allows incorporation of model uncertainties in the control design. The optimal SOF– $\mathcal{H}_\infty$  problem can be formally stated in the following term: *Find a SOF matrix  $F$  such that  $A(F)$  is Hurwitz and the  $\mathcal{H}_\infty$ -norm of (26.26) is minimal.* We consider the following well known  $\mathcal{H}_\infty$ -BMI version, see, e.g. [20]:

$$\begin{aligned} \min \quad & \gamma \quad \text{s.t. } X > 0, \quad \gamma > 0, \\ \left[ \begin{array}{ccc} A(F)^T X + X A(F) & X B(F) & C(F)^T \\ B(F)^T X & -\gamma I_{n_w} & D(F)^T \\ C(F) & D(F) & -\gamma I_{n_z} \end{array} \right] < 0, \end{aligned} \tag{26.28}$$

where  $\gamma \in \mathbb{R}$ ,  $X \in \mathbb{R}^{n_x \times n_x}$ .

We present results of our numerical experiences for the static output feedback problems of *COMPlib*. The link between *COMPlib* and PENBMI was provided by the MATLAB parser YALMIP 3 [22]. All tests were performed on a 2.5 GHz Pentium with 1 GB RDRAM under Linux. The results of PENBMI for  $\mathcal{H}_2$ -BMI and  $\mathcal{H}_{\infty}$ -BMI problems can be divided into seven groups: The first group consists of examples solved without any difficulties (38 problems in the  $\mathcal{H}_2$  case and 37 problems for the  $\mathcal{H}_{\infty}$  setting). The second and third group contain all cases for which we had to relax our stopping criterion. In 4 (11) examples the achieved precision was still close to our predefined stopping criterion, while in 5 (7) cases deviation is significant (referring to  $\mathcal{H}_2$  ( $\mathcal{H}_{\infty}$ )). Then there are examples, for which we could calculate almost feasible solutions, but which failed to satisfy the Hurwitz-criterion, namely AC5 and NN10. The fourth and fifth group consist of medium and small scale cases for which PENBMI failed, due to ill conditioned Hessian of  $F$  – the Cholesky algorithm used for its factorization did not deliver accurate solution and the Newton method failed. In the  $\mathcal{H}_2$ -setting (fourth group) these are AC7, AC9, AC13, AC18, JE1, JE2, JE3, REA4, DIS5, WEC1, WEC2, WEC3, UWV, PAS, NN1, NN3, NN5, NN6, NN7, NN9, NN12 and NN17, in the  $\mathcal{H}_{\infty}$ -setting (fifth group) JE1, JE2, JE3, REA4, DIS5, UWV, PAS, TF3, NN1, NN3, NN5, NN6, NN7 and NN13. The cases in the sixth group are large scale, ill conditioned problems, where PENBMI ran out of time (AC10, AC14, CSE2, EB5). Finally, for very large test cases our code runs out of memory (HS1, BDT2, EB6, TL, CDP, NN18).

### 26.4.3 Simultaneous Stabilization BMIs

Another example leading to BMI formulation is the problem of simultaneously stabilizing a family of single-input single-output linear systems by one fixed controller of given order. This problem arises for instance when trying to preserve stability of a control system under the failure of sensors, actuators, or processors. Simultaneous stabilization of three or more systems was extensively studied in [2]. Later on, the problem was shown to belong to the wide range of robust control problems that are NP-hard [3].

In [14] a BMI formulation of the simultaneous stabilization problem was obtained in the framework of the polynomial, or algebraic approach to systems

control. This formulation leads to a feasibility BMI problem which, in a more general setting can be reformulated by the following procedure: Assume we want to find a feasible point of the following system of BMIs

$$A_0^i + \sum_{k=1}^n x_k A_k^i + \sum_{k=1}^n \sum_{\ell=1}^n x_k x_\ell K_{k\ell}^i < 0, \quad i = 1, \dots, N \quad (26.29)$$

with symmetric matrices  $A_k^i, K_{k\ell}^i \in \mathbb{R}^{d_i \times d_i}$ ,  $k, \ell = 1, \dots, n$ ,  $i = 1, \dots, N$ , and  $x \in \mathbb{R}^n$ . Then we can check the feasibility of (26.29) by solving the following optimization problem

$$\min_{x \in \mathbb{R}^n, \lambda \in \mathbb{R}} \lambda \quad (26.30)$$

$$\text{s.t.} \quad A_0^i + \sum_{k=1}^n x_k A_k^i + \sum_{k=1}^n \sum_{\ell=1}^n x_k x_\ell K_{k\ell}^i \leq \lambda I_n, \quad i = 1, \dots, N. \quad (26.31)$$

Problem (26.30) is a global optimization problem: we know that if its global minimum  $\lambda$  is non-negative then the original problem (26.29) is infeasible. On the other hand PENBMI can only find critical points, so when solving (26.30), the only conclusion we can make is the following:

- when  $\lambda < 0$ , the system is strictly feasible;
- when  $\lambda = 0$ , the system is marginally feasible;
- when  $\lambda > 0$  the system may be infeasible.

During numerical experiments it turned out that the feasible region of (26.29) is often unbounded. We used two strategies to avoid numerical difficulties in this case: First we introduced large enough artificial bounds  $x_{\text{bound}}$ . Second, we modify the objective function by adding the square of the 2-norm of the vector  $x$  multiplied by a weighting parameter  $w$ . After these modifications problem (26.30) reads as follows:

$$\begin{aligned} & \min_{x \in \mathbb{R}^n, \lambda \in \mathbb{R}} \lambda + w \|x\|_2^2 \\ \text{s.t.} \quad & -x_{\text{bound}} \leq x^k \leq x_{\text{bound}}, \quad k = 1, \dots, n \\ & A_0^i + \sum_{k=1}^n x_k A_k^i + \sum_{k=1}^n \sum_{\ell=1}^n x_k x_\ell K_{k\ell}^i \leq \lambda I_n, \quad i = 1, \dots, N. \end{aligned} \quad (26.32)$$

This is exactly the problem formulation we used in our numerical experiments.

Results of numerical examples for a suite of simultaneous stabilization problems selected from the recent literature can be found in [13].

## 26.5 PENNON

### 26.5.1 The Problem and the Modified Algorithm

#### 26.5.1.1 Problem Formulation

In this, so far most general version of the code, we solve optimization problems with a nonlinear objective subject to nonlinear inequality and equality constraints and semidefinite bound constraints:

$$\begin{aligned} & \min_{x \in \mathbb{R}^n, Y_1 \in \mathbb{S}^{p_1}, \dots, Y_k \in \mathbb{S}^{p_k}} f(x, Y) \\ \text{subject to } & g_i(x, Y) \leq 0, \quad i = 1, \dots, m_g \\ & h_i(x, Y) = 0, \quad i = 1, \dots, m_h \\ & \underline{\lambda}_i I \leq Y_i \leq \bar{\lambda}_i I, \quad i = 1, \dots, k. \end{aligned} \quad (26.33)$$

Here

- $x \in \mathbb{R}^n$  is the vector variable
- $Y_1 \in \mathbb{S}^{p_1}, \dots, Y_k \in \mathbb{S}^{p_k}$  are the matrix variables; we denote  $Y = (Y_1, \dots, Y_k)$
- $f, g_i$  and  $h_i$  are  $C^2$  functions from  $\mathbb{R}^n \times \mathbb{S}^{p_1} \times \dots \times \mathbb{S}^{p_k}$  to  $\mathbb{R}$
- $\underline{\lambda}_i$  and  $\bar{\lambda}_i$  are the lower and upper bounds, respectively, on the eigenvalues of  $Y_i$ ,  $i = 1, \dots, k$

Although the semidefinite inequality constraints are of a simple type, most nonlinear SDP problems can be formulated in the above form. For instance, the problem (26.1) can be transformed into (26.33) using slack variables and equality constraints, when

$$\mathcal{A}(x) \leq 0$$

is replaced by

$$\mathcal{A}(x) = S \quad \text{element-wise}$$

$$S \leq 0$$

with a new matrix variable  $S \in \mathbb{S}^m$ .

#### 26.5.1.2 Direct Equality Handling

Problem (26.33) is not actually a problem of type (26.1) that was introduced in the first section and for which we have developed the convergence theory. The new element here are the equality constraints. Of course, we can formulate the

equalities as two inequalities, and this works surprisingly well for many problems. However, to treat the equalities in a “proper” way, we adopted a concept which is successfully used in modern primal-dual interior point algorithms (see, e.g., [32]): rather than using augmented Lagrangians, we handle the equality constraints directly on the level of the subproblem. This leads to the following approach. Consider the optimization problem

$$\begin{aligned} & \min_{x \in \mathbb{R}^n} f(x) \\ & \text{subject to} \\ & \quad \mathcal{A}(x) \leq 0, \\ & \quad h(x) = 0, \end{aligned} \tag{26.34}$$

where  $f$  and  $\mathcal{A}$  are defined as in the previous sections and  $h : \mathbb{R}^n \rightarrow \mathbb{R}^d$  represents a set of equality constraints. Then we define the augmented Lagrangian

$$\bar{F}(x, U, v, p) = f(x) + \langle U, \Phi_p(\mathcal{A}(x)) \rangle_{\mathbb{S}^m} + v^\top h(x), \tag{26.35}$$

where  $U, \Phi, p$  are defined as before and  $v \in \mathbb{R}^d$  is the vector of Lagrangian multipliers associated with the equality constraints. Now, on the level of the subproblem, we attempt to find an approximate solution of the following system (in  $x$  and  $v$ ):

$$\begin{aligned} \nabla_x \bar{F}(x, U, v, p) &= 0, \\ h(x) &= 0, \end{aligned} \tag{26.36}$$

where the penalty parameter  $p$  as well as the multiplier  $U$  are fixed. In order to solve systems of type (26.36), we apply the damped Newton method. Descent directions are calculated utilizing the factorization routine MA27 from the Harwell subroutine library ([6]) in combination with an inertia correction strategy as described in [32]. Moreover, the step length is derived using an augmented Lagrangian merit function defined as

$$\bar{F}(x, U, v, p) + \frac{1}{2\mu} \|h(x)\|_2^2$$

along with an Armijo rule.

### 26.5.1.3 Strictly Feasible Constraints

In certain applications, the bound constraints must remain strictly feasible for all iterations because, for instance, the objective function may be undefined at infeasible points [17]. To be able to solve such problems, we treat these inequalities

by a classic barrier function. For this reason we introduce an additional matrix inequality

$$\mathcal{S}(x) \leq 0$$

in problem (26.2) and define the augmented Lagrangian

$$\tilde{F}(x, U, p, s) = f(x) + \langle U, \Phi_p(\mathcal{A}(x)) \rangle_{\mathbb{S}^m} + s\Phi_{\text{bar}}(\mathcal{S}(x)), \quad (26.37)$$

where  $\Phi_{\text{bar}}$  can be defined, for example, by

$$\Phi_{\text{bar}}(\mathcal{S}(x)) = -\log \det(-\mathcal{S}(x)).$$

Note that, while the penalty parameter  $p$  maybe constant from a certain index  $\bar{k}$  (see again [31] for details), the barrier parameter  $s$  is required to tend to zero with increasing  $k$ .

## 26.5.2 The Code PENNON

### 26.5.2.1 Slack Removal

As already mentioned, to transform constraints of the type  $\mathcal{A}(x) \leq 0$  into our standard structure, we need to introduce a slack matrix variable  $S$ , and replace the original constraint by  $\mathcal{A}(x) = S$  element-wise, and  $S \leq 0$ . Thus in order to formulate the problem in the required form, we have to introduce a new (possibly large) matrix variable and many new equality constraints, which may have a negative effect on the performance of the algorithm. However, the reformulation using slack variables is only needed for the input of the problem, not for its solution by Algorithm 26.1. Hence, the user has the option to say that certain matrix variables are actually slacks and these are then automatically removed by a preprocessor. The code then solves the problem with the original constraint  $\mathcal{A}(x) \leq 0$ .

### 26.5.2.2 User Interface

Unlike in the PENSDP and PENBMI case, the user has to provide not only function values but also the first and second derivatives of the objective and constraint functions. In the MATLAB and C/C++/Fortran interface the user is required to provide six functions/subroutines for evaluation of function value, gradient and Hessian of the objective function and the constraints, respectively, at a given point.

To make things simple, the matrix variables are treated as vectors in these functions, using the operator  $\text{svec} : \mathbb{S}^m \rightarrow \mathbb{R}^{(m+1)m/2}$  defined by

$$\text{svec} \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ & a_{22} & \dots & a_{2m} \\ & \ddots & \vdots \\ \text{sym} & & & a_{mm} \end{pmatrix} = (a_{11}, a_{12}, a_{22}, \dots, a_{1m}, a_{2m}, \dots, a_{mm})^T$$

In the main program, the user defines problem sizes, values of bounds, and information about matrix variables (number, sizes and sparsity patterns).

In addition, we also provide an interface to AMPL [7] which is a comfortable modeling language for optimization problems. As AMPL does not support matrix variables, we treat them, within an AMPL script, as vectors, using the operator svec defined above.

*Example 26.1.* Assume that we have a matrix variable  $X \in \mathbb{S}^3$

$$X = \begin{pmatrix} x_1 & x_2 & x_4 \\ x_2 & x_3 & x_5 \\ x_4 & x_5 & x_6 \end{pmatrix}$$

and a constraint

$$\text{Tr}(XA) = 3 \quad \text{with } A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

The matrix variable is treated as a vector

$$\text{svec}(X) = (x_1, x_2, \dots, x_6)^T$$

and the above constraint is thus equivalent to

$$x_3 + 2x_4 = 3.$$

The code needs to identify the matrix variables, their number and size. These data are included in an ASCII file that is directly read by the PENNON and that, in addition, includes information about lower and upper bounds on these variables.

### 26.5.3 Examples

Most examples of nonlinear semidefinite programs that can be found in the literature are of the form: for a given (symmetric, indefinite) matrix  $H$  find a nearest positive semidefinite matrix satisfying possibly some additional constraints. Many of these problems can be written as follows

$$\min_{X \in \mathbb{S}^n} \frac{1}{2} \|X - H\|_F^2$$

subject to

$$\langle A_i, X \rangle = b_i, \quad i = 1, \dots, m$$

$$X \geq 0 \tag{26.38}$$

with  $A_i \in \mathbb{S}^n$ ,  $i = 1, \dots, m$ . Probably the most prominent example is the problem of finding the nearest correlation matrix [15].

Several algorithms have been derived for the solution of this problems; see, e.g., [15, 23]. It is not our primal goal to compete with these specialized algorithms (although PENNON can solve problems of type (26.38) rather efficiently). Rather we want to utilize the full potential of our code and solve “truly nonlinear” semidefinite problems. In the rest of this section we will give examples of such problems.

### 26.5.4 Correlation Matrix with the Constrained Condition Number

We consider the problem of finding the nearest correlation matrix:

$$\min_X \sum_{i,j=1}^n (X_{ij} - H_{ij})^2$$

subject to

$$X_{ii} = 1, \quad i = 1, \dots, n$$

$$X \geq 0 \tag{26.39}$$

We will consider an example based on a practical application from finances; see [33]. Assume that a  $5 \times 5$  correlation matrix is extended by one row and column. The new data is based on a different frequency than the original part of the matrix, which means that the new matrix is no longer positive definite:

$$H_{\text{ext}} = \begin{pmatrix} 1 & -0.44 & -0.20 & 0.81 & -0.46 & -0.05 \\ -0.44 & 1 & 0.87 & -0.38 & 0.81 & -0.58 \\ -0.20 & 0.87 & 1 & -0.17 & 0.65 & -0.56 \\ 0.81 & -0.38 & -0.17 & 1 & -0.37 & -0.15 \\ -0.46 & 0.81 & 0.65 & -0.37 & 1 & -0.08 \\ -0.05 & -0.58 & -0.56 & -0.15 & 0.08 & 1 \end{pmatrix}.$$

Let us find the nearest correlation matrix to  $H_{\text{ext}}$  by solving (26.39) (either by PENNON or by any of the specialized algorithms mentioned at the beginning of this section). We obtain the following result (for the presentation of results, we will use MATLAB output in short precision):

$$\begin{aligned} \mathbf{X} = \\ \begin{array}{cccccc} \mathbf{1.0000} & -\mathbf{0.4420} & -\mathbf{0.2000} & \mathbf{0.8096} & -\mathbf{0.4585} & -\mathbf{0.0513} \\ -\mathbf{0.4420} & \mathbf{1.0000} & \mathbf{0.8704} & -\mathbf{0.3714} & \mathbf{0.7798} & -\mathbf{0.5549} \\ -\mathbf{0.2000} & \mathbf{0.8704} & \mathbf{1.0000} & -\mathbf{0.1699} & \mathbf{0.6497} & -\mathbf{0.5597} \\ \mathbf{0.8096} & -\mathbf{0.3714} & -\mathbf{0.1699} & \mathbf{1.0000} & -\mathbf{0.3766} & -\mathbf{0.1445} \\ -\mathbf{0.4585} & \mathbf{0.7798} & \mathbf{0.6497} & -\mathbf{0.3766} & \mathbf{1.0000} & \mathbf{0.0608} \\ -\mathbf{0.0513} & -\mathbf{0.5549} & -\mathbf{0.5597} & -\mathbf{0.1445} & \mathbf{0.0608} & \mathbf{1.0000} \end{array} \end{aligned}$$

with eigenvalues

$$\begin{aligned} \mathbf{eigen} = \\ \begin{array}{cccccc} \mathbf{0.0000} & \mathbf{0.1163} & \mathbf{0.2120} & \mathbf{0.7827} & \mathbf{1.7132} & \mathbf{3.1757} \end{array} \end{aligned}$$

As we can see, one eigenvalue of the nearest correlation matrix is zero. This is highly undesirable from the application point of view. To avoid this, we can add lower (and upper) bounds on the matrix variable, i.e., constraints  $\underline{\lambda}I \leq X \leq \bar{\lambda}I$ . However, the application requires a different approach when we need to *bound the condition number of the nearest correlation matrix*, i.e., to add the constraint

$$\text{cond}(X) \leq \kappa.$$

This constraint can be introduced in several ways. For instance, we can introduce the constraint

$$I \leq \widetilde{X} \leq \kappa I$$

using the transformation  $\widetilde{X} = \zeta X$ . The problem of finding the nearest correlation matrix with a *given* condition number then reads as follows:

$$\min_{\zeta, \widetilde{X}} \sum_{i,j=1}^n \left( \frac{1}{\zeta} \widetilde{X}_{ij} - H_{ij} \right)^2$$

subject to

$$\widetilde{X}_{ii} - \zeta = 0, \quad i = 1, \dots, n$$

$$I \leq \widetilde{X} \leq \kappa I. \tag{26.40}$$

The new problem now has the NLP-SDP structure of (26.33). When solving it by PENNON with  $\kappa = 10$ , we get the solution after 11 outer and 37 inner iterations. The optimal value of  $\zeta$  is 3.4886 and, after the back substitution  $X = \frac{1}{\zeta} \widetilde{X}$ , we get the nearest correlation matrix

```
X =
 1.0000 -0.3775 -0.2230  0.7098 -0.4272 -0.0704
 -0.3775 1.0000  0.6930 -0.3155  0.5998 -0.4218
 -0.2230  0.6930 1.0000 -0.1546  0.5523 -0.4914
 0.7098 -0.3155 -0.1546 1.0000 -0.3857 -0.1294
 -0.4272  0.5998  0.5523 -0.3857  1.0000 -0.0576
 -0.0704 -0.4218 -0.4914 -0.1294 -0.0576 1.0000
```

with eigenvalues

```
eigenvals =
 0.2866    0.2866    0.2867    0.6717    1.6019    2.8664
```

and the condition number equal to 10, indeed.

#### 26.5.4.1 Large-Scale Problems

To test the capability of the code to solve large-scale problems, we have generated randomly perturbed correlation matrices  $H$  of arbitrary dimension by the commands

```
n = 500; x=10.^[-4:4/(n-1):0];
G = gallery('randcorr',n*x/sum(x));
E = 2*rand(n,n)-ones(n,n); E=triu(E)+triu(E,1)';
E=(E+E')/2;
H = (1-0.1).*G + 0.1*E;
```

For large-scale problems, we successfully use the iterative (preconditioned conjugate gradient) solver for the Newton system in Step 1 of the algorithm. In every Newton step, the iterative solver needs just a few iterations, making it a very efficient alternative to a direct solver.

For instance, to solve a problem with a  $500 \times 500$  matrix  $H$  we needed 11 outer and 148 inner iterations, 962 CG steps, and 21 min on a notebook. Note that the problem had 125,251 variables and 500 linear constraints: that means that at each Newton step we solved (approximately) a system with a *full*  $125,251 \times 125,251$  matrix. The iterative solver (needing just matrix-vector product) was clearly the only alternative here.

We have also successfully solved a real-world problem with a matrix of dimension 2,000 and with many additional linear constraints in about 10 h on a standard Linux workstation with four Intel Core 2 Quad processors with 2.83 GHz and 8 Gbyte of memory (using only one processor).

#### 26.5.5 Approximation by Nonnegative Splines

Consider the problem of approximating a one-dimensional function given only by a large amount of noisy measurement by a cubic spline. Additionally, we require

that the function is nonnegative. This kind of problem arises in many application, for instance, in shape optimization considering unilateral contact or in arrival rate approximation [1].

Assume that function  $f : \mathbb{R} \rightarrow \mathbb{R}$  is defined on interval  $[0, 1]$ . We are given its function values  $b_i$ ,  $i = 1, \dots, n$  at points  $t_i \in (0, 1)$ . We may further assume that the function values are subject to a random noise. We want to find a smooth approximation of  $f$  by a cubic spline, i.e., by a function of the form

$$P(t) = P^{(i)}(t) = \sum_{k=0}^3 P_k^{(i)}(t - a_{i-1})^k \quad (26.41)$$

for a point  $t \in [a_{i-1}, a_i]$ , where  $0 = a_0 < a_1 < \dots < a_m = 1$  are the knots and  $P_k^{(i)}$  ( $i = 1, \dots, m$ ,  $k = 0, 1, 2, 3$ ) the coefficients of the spline. The spline property that  $P$  should be continuous and have continuous first and second derivatives is expressed by the following equalities for  $i = 1, \dots, m-1$ :

$$P_0^{(i+1)} - P_0^{(i)} - P_1^{(i)}(a_i - a_{i-1}) - P_2^{(i)}(a_i - a_{i-1})^2 - P_3^{(i)}(a_i - a_{i-1})^3 = 0 \quad (26.42)$$

$$P_1^{(i+1)} - P_1^{(i)} - 2P_2^{(i)}(a_i - a_{i-1}) - 3P_3^{(i)}(a_i - a_{i-1})^2 = 0 \quad (26.43)$$

$$2P_2^{(i+1)} - 2P_2^{(i)} - 6P_3^{(i)}(a_i - a_{i-1}) = 0. \quad (26.44)$$

The function  $f$  will be approximated by  $P$  in the least square sense, so we want to minimize

$$\sum_{j=1}^n (P(t_j) - b_j)^2$$

subject to (26.42)–(26.44).

Now, the original function  $f$  is assumed to be nonnegative and we also want the approximation  $P$  to have this property. A simple way to guarantee nonnegativity of a spline is to express it using  $B$ -splines and consider only nonnegative  $B$ -spline coefficients. However, it was shown by de Boor and Daniel [5] that this may lead to a poor approximation of  $f$ . In particular, they showed that while approximation of a nonnegative function by nonnegative splines of order  $k$  gives errors of order  $h^k$ , approximation by a subclass of nonnegative splines of order  $k$  consisting of all those whose  $B$ -spline coefficients are nonnegative may yield only errors of order  $h^2$ . In order to get the best possible approximation, we use a result by Nesterov [26] saying that  $P^{(i)}(t)$  from (26.41) is nonnegative if and only if there exist two symmetric matrices

$$X^{(i)} = \begin{pmatrix} x_i & y_i \\ y_i & z_i \end{pmatrix}, \quad S^{(i)} = \begin{pmatrix} s_i & v_i \\ v_i & w_i \end{pmatrix}$$

such that

$$P_0^{(i)} = (a_i - a_{i-1})s_i \quad (26.45)$$

$$P_1^{(i)} = x_i - s_i + 2(a_i - a_{i-1})v_i \quad (26.46)$$

$$P_2^{(i)} = 2y_i - 2v_i + (a_i - a_{i-1})w_i \quad (26.47)$$

$$P_3^{(i)} = z_i - w_i \quad (26.48)$$

$$X^{(i)} \geq 0, \quad S^{(i)} \geq 0. \quad (26.49)$$

Summarizing, we want to solve an NLP-SDP problem

$$\min_{\substack{P_k^{(i)} \in \mathbb{R} \\ i=1,\dots,m, k=0,1,2,3}} \sum_{j=1}^n (P(t_j) - b_j)^2$$

subject to

$$\begin{aligned} & (26.42), (26.43), (26.44), \quad i = 1, \dots, m \\ & (26.45) - (26.49), \quad i = 1, \dots, m. \end{aligned} \quad (26.50)$$

More complicated (“more nonlinear”) objective functions can be obtained when considering, for instance, the problem of approximating the arrival rate function of a non-homogeneous Poisson process based on observed arrival data [1].

*Example 26.2.* A problem of approximating a cosine function given at 500 points by noisy data of the form  $\cos(4*\pi*rand(500, 1))+1+.5.*rand(500, 1)-.25$  approximated by a nonnegative cubic spline with 7 knots lead to an NSDP problem in 80 variables, 16 matrix variables, 16 matrix constraints, and 49 linear inequality constraints. The problem was solved by PENNON in about 1 s using 17 global and 93 Newton iterations.

**Acknowledgements** The authors would like to thank Didier Henrion and Johan Löfgren for their constant help during the code development. The work has been partly supported by grant A100750802 of the Czech Academy of Sciences (MK) and by DFG cluster of excellence 315 (MS). The manuscript was finished while the first author was visiting the Institute for Pure and Applied Mathematics, UCLA. The support and friendly atmosphere of the Institute are acknowledged with gratitude.

## References

1. Alizadeh, F., Eckstein, J., Noyan, N., Rudolf, G.: Arrival rate approximation by nonnegative cubic splines. Operations Research **56**, 140–156 (2008)
2. Blondel, V.D.: Simultaneous stabilization of linear systems. MacMillan, New York (1994)

3. Blondel, V.D., Tsitsiklis, J.N.: A survey of computational complexity results in systems and control. *Automatica* **36**(9), 1249–1274 (2000)
4. Bonnans, F.J., Shapiro, A.: Perturbation Analysis of Optimization Problems. Springer-Verlag New-York (2000)
5. de Boor, C., Daniel, J.W.: Splines with nonnegative  $b$ -spline coefficients. *Math. Comp.* **28**(4-5), 565–568 (1974)
6. Duff, I.S., Reid, J.K.: MA27—A set of Fortran subroutines for solving sparse symmetric sets of linear equations. Tech. Report R.10533, AERE, Harwell, Oxfordshire, UK (1982)
7. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL: A Modeling Language for Mathematical Programming. The Scientific Press (1993)
8. Fujisawa, K., Kojima, M., Nakata, K.: Exploiting sparsity in primal-dual interior-point method for semidefinite programming. *Mathematical Programming* **79**, 235–253 (1997)
9. Fujisawa, K., Kojima, M., Nakata, K., Yamashita, M.: SDPA User’s Manual—Version 6.00. Technical report, Department of Mathematical and Computing Science, Tokyo University of Technology (2002)
10. Fukuda, M., Kojima, M., Shida, M.: Lagrangian dual interior-point methods for semidefinite programs. *SIAM J. Optimization* **12**, 1007–1031 (2002)
11. Geiger, C., Kanzow, C.: Numerische Verfahren zur Lösung unrestrictierter Optimierungsaufgaben. Springer-Verlag (1999). In German.
12. Goh, K.C., Turan, L., Safonov, M.G., Papavassilopoulos, G.P., Ly, J.H.: Biaffine matrix inequality properties and computational methods. In: Proceedings of the American Control Conference, Baltimore, MD (1994)
13. Henrion, D., Kočvara, M., Stingl, M.: Solving simultaneous stabilization bmi problems with pennon. LAAS-CNRS research report no. 04508, LAAS, Toulouse (2003)
14. Henrion, D., Tarbouriech, S., Šebek, M.: Rank-one LMI approach to simultaneous stabilization of linear systems. *Systems and control letters* **38**(2), 79–89 (1999)
15. Higham, N.J.: Computing the nearest correlation matrix—A problem from finance. *IMA J. Numer. Anal.* **22**(3), 329–343 (2002)
16. Kočvara, M., Stingl, M.: PENNON—a code for convex nonlinear and semidefinite programming. *Optimization Methods and Software* **18**(3), 317–333 (2003)
17. Kočvara, M., Stingl, M.: Free material optimization: Towards the stress constraints. *Structural and Multidisciplinary Optimization* **33**(4-5), 323–335 (2007)
18. Kočvara, M., Stingl, M.: On the solution of large-scale SDP problems by the modified barrier method using iterative solvers. *Mathematical Programming (Series B)* **109**(2-3), 413–444 (2007)
19. Kočvara, M., Stingl, M.: On the solution of large-scale SDP problems by the modified barrier method using iterative solvers: Erratum. *Mathematical Programming (Series B)* **120**(1), 285–287 (2009)
20. Leibfritz, F.: COMPlib: COnstrained Matrix-optimization Problem library – a collection of test examples for nonlinear semidefinite programs, control system design and related problems. Technical report, University of Trier, Department of Mathematics, D–54286 Trier, Germany (2003)
21. Li, Y., Zhang, L.: A new nonlinear lagrangian method for nonconvex semidefinite programming. *Journal of Applied Analysis* **15**(2), 149–172 (2009)
22. Löfberg, J.: YALMIP : A toolbox for modeling and optimization in MATLAB. In: Proceedings of the CACSD Conference, Taipei, Taiwan (2004)
23. Malick, J.: A dual approach to semidefinite least-squares problems. *SIAM J. Matrix Analysis and Applications* **26**(1), 272–284 (2005)
24. Mittelmann, H.D.: An independent benchmarking of SDP and SOCP solvers. *Math. Prog.* **95**, 407–430 (2003)
25. Morales, J.L., Nocedal, J.: Automatic preconditioning by limited memory quasi-Newton updating. *SIAM Journal on Optimization* **10**, 1079–1096 (2000)

26. Nesterov, Y.: Squared functional systems and optimization problems. In: Frenk, H., Roos, K., Terlaky, T. (eds.) High performance optimization (Chapter 17), pp. 405–440. Kluwer Academic Publishers, Dordrecht (2000)
27. Ng, E., Peyton, B.W.: Block sparse cholesky algorithms on advanced uniprocessor computers. SIAM Journal on Scientific Computing **14**, 1034–1056 (1993)
28. Nocedal, J., Wright, S.: Numerical Optimization. Springer Series in Operations Research. Springer, New York (1999)
29. Noll, D.: Local convergence of an augmented lagrangian method for matrix inequality constrained programming. Optimization Methods and Software **22**(5), 777–802 (2007)
30. Polyak, R.: Modified barrier functions: Theory and methods. Mathematical Programming **54**, 177–222 (1992)
31. Stingl, M.: On the Solution of Nonlinear Semidefinite Programs by Augmented Lagrangian Methods. PhD thesis, Institute of Applied Mathematics II, Friedrich-Alexander University of Erlangen-Nuremberg (2006)
32. Wächter, A., Biegler, L.T.: On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. Math. Prog. **106**, 25–57 (2006)
33. Werner, R., Schöttle, K.: Calibration of correlation matrices—SDP or not SDP. Preprint available from [www.gloriamundi.org](http://www.gloriamundi.org) (2007)
34. Wright, S.: Primal-Dual Interior-Point Methods. SIAM, Philadelphia, PA (1997)

## **Part IV**

# **Applications**

# Chapter 27

## SDP Relaxations for Some Combinatorial Optimization Problems

Renata Sotirov

### 27.1

In this chapter we present recent developments on solving various combinatorial optimization problems by using semidefinite programming (SDP). We present several SDP relaxations of the quadratic assignment problem and the traveling salesman problem. Further, we show the equivalence of several known SDP relaxations of the graph equipartition problem, and present recent results on the bandwidth problem.

#### Notation

The space of  $p \times q$  real matrices is denoted by  $\mathbb{R}^{p \times q}$ , the space of  $k \times k$  symmetric matrices is denoted by  $\mathcal{S}_k$ , and the space of  $k \times k$  symmetric positive semidefinite matrices by  $\mathcal{S}_k^+$ . We will sometimes also use the notation  $X \geq 0$  instead of  $X \in \mathcal{S}_k^+$ , if the order of the matrix is clear from the context.

For index sets  $\alpha, \beta \subset \{1, \dots, n\}$ , we denote the submatrix that contains the rows of  $A$  indexed by  $\alpha$  and the columns indexed by  $\beta$  as  $A(\alpha, \beta)$ . If  $\alpha = \beta$ , the principal submatrix  $A(\alpha, \alpha)$  of  $A$  is abbreviated as  $A(\alpha)$ . The  $i$ th column of a matrix  $C$  is denoted by  $C_{:,i}$ .

We use  $I_n$  to denote the identity matrix of order  $n$ , and  $e_i$  to denote the  $i$ -th standard basis vector. Similarly,  $J_n$  and  $u_n$  denote the  $n \times n$  all-ones matrix and all-ones  $n$ -vector respectively, and  $0_{n \times n}$  is the zero matrix of order  $n$ . We will omit subscripts if the order is clear from the context. The set of  $n \times n$  permutation matrices is denoted by  $\Pi_n$ . We set  $E_{ij} = e_i e_j^T$ .

---

R. Sotirov (✉)

Department of Econometrics and Operations Research, Tilburg University,  
Warandelaan 2, 5000 LE Tilburg, The Netherlands

e-mail: [rsotirov@uvt.nl](mailto:rsotirov@uvt.nl)

The “vec” operator stacks the columns of a matrix, while the “diag” operator maps an  $n \times n$  matrix to the  $n$ -vector given by its diagonal. The adjoint operator of “diag” we denote by “Diag”. The trace operator is denoted by “tr”.

The Kronecker product  $A \otimes B$  of matrices  $A \in \mathbb{R}^{p \times q}$  and  $B \in \mathbb{R}^{r \times s}$  is defined as the  $pr \times qs$  matrix composed of  $pq$  blocks of size  $r \times s$ , with block  $ij$  given by  $a_{ij}B$ ,  $i = 1, \dots, p$ ,  $j = 1, \dots, q$ .

## 27.2 The Quadratic Assignment Problem

The quadratic assignment problem (QAP) was introduced in 1957 by Koopmans and Beckmann as a mathematical model for assigning a set of economic activities to a set of locations. Nowadays, the QAP is widely considered as a classical combinatorial optimization problem. The QAP is also known as a generic model for various real-life problems, such as hospital layout [27, 48], balancing of turbine runners [49], decoding for multi-input multi-output systems [54, 55], etc. For a more detailed list on applications of the QAP see e.g., [11, 12, 15].

We study the Koopmans–Beckmann form of the QAP, which can be written as

$$(QAP) \quad \begin{aligned} & \min \operatorname{tr}(AXB + C)X^T \\ & \text{s.t. } X \in \Pi_n, \end{aligned} \tag{27.1}$$

where  $A$  and  $B$  are given symmetric  $n \times n$  matrices, and  $C$  is a square matrix of order  $n$ . It is well-known that the QAP contains the traveling salesman problem as a special case and is therefore NP-hard in the strong sense. Moreover, experience has shown that instances with  $n = 30$  are already very hard to solve in practice. Thus it is typically necessary to use massive parallel computing to solve even moderately sized QAP instances.

The recent developments in algorithms as well as in computational platforms have resulted in a large improvement in the capability to solve QAPs exactly. Anstreicher et al. [4] made a break-through by solving a number of previously unsolved large QAPs from QAPLIB [10]. They solved to optimality several famous instances, including the problem of size  $n = 30$  posed by Nugent et al. [56], and problems of size  $n = 30, 32$  posed by Krarup and Pruzan [47]. Optimal solutions of the problems of size  $n = 36$  posed by Steinberg [65] are reported in [8, 57]. For a detailed survey on recent developments regarding the QAP problem, see Anstreicher [2].

In [4], the authors incorporate a convex quadratic programming bound that was introduced by Anstreicher and Brixius in [3], into a branch and bound framework that was running on a computational grid. Their computations are considered to be among the most extensive computations ever performed to solve combinatorial optimization problems. The computational work to solve a problem of size  $n = 30$  (in particular Nug30) took the equivalent of nearly 7 years of computation time on a single HP9000 C3000 workstation.

In the sequel we describe the quadratic programming bound (QPB) that is introduced in [3]. Let  $V$  be an  $n \times (n-1)$  matrix whose columns form an orthonormal basis for the nullspace of  $u_n^T$ , and define  $\hat{A} := V^T A V$ ,  $\hat{B} := V^T B V$ . The QPB bound is:

$$\begin{aligned} & \min \operatorname{vec}(X)^T Q \operatorname{vec}(X) + \operatorname{tr}(CX^T) + \sum_{i=1}^n \lambda(\hat{A})_i \lambda(\hat{B})_{n+1-i} \\ (\text{QPB}) \quad & \text{s.t. } Xu_n = X^T u_n = u_n \\ & X \geq 0, \end{aligned}$$

where  $Q := (B \otimes A) - (I \otimes V \hat{S} V^T) - (V \hat{T} V^T \otimes I)$ , and  $\lambda(\cdot)_1 \leq \dots \leq \lambda(\cdot)_n$  are the eigenvalues of the matrix under consideration. The matrices  $\hat{S}$  and  $\hat{T}$  are optimal solutions of the following SDP problem

$$\begin{aligned} \text{OPT}_{\text{SDD}_{(\hat{A}, \hat{B})}} := & \max \operatorname{tr}(\hat{S}) + \operatorname{tr}(\hat{T}) \\ \text{s.t. } & (\hat{B} \otimes \hat{A}) - (I \otimes \hat{S}) - (\hat{T} \otimes I) \succeq 0. \end{aligned}$$

It is shown by Anstreicher and Wolkowicz [5] that

$$\text{OPT}_{\text{SDD}_{(\hat{A}, \hat{B})}} = \sum_{i=1}^n \lambda(\hat{A})_i \lambda(\hat{B})_{n+1-i},$$

and in [3] that optimal  $\hat{S}$ ,  $\hat{T}$  can be obtained from the spectral decomposition of  $\hat{A}$  and  $\hat{B}$ . The objective in QPB is convex on the nullspace of the constraints  $Xu = X^T u = u$ , or equivalently  $(V^T \otimes V^T)Q(V \otimes V) \succeq 0$ . Therefore, QPB is a convex quadratic programming program.

In [3], the quadratic programming bounds are computed using a long-step path following interior point algorithm. However, it turned out to be too costly to use interior point methods (IPMs) in a branch and bound setting, particularly for the large scale problems. Therefore, in [4, 7] the Frank–Wolfe (FW) algorithm [30] is implemented to approximately solve QPB in each node of the branching tree. Although the FW method is known to have poor asymptotic performance, it seems to be a good choice in the mentioned branch and bound setting. Namely, each iteration of the FW algorithm requires the solution of a linear assignment problem that can be performed extremely rapidly, and the FW algorithm generates dual information that can be exploited while branching.

Another class of convex relaxations for the QAP are the semidefinite programming bounds. The following SDP relaxation of the QAP was studied by Povh and Rendl [60]:

$$\begin{aligned} & \min \operatorname{tr}(B \otimes A + \operatorname{Diag}(\operatorname{vec}(C)))Y \\ & \text{s.t. } \operatorname{tr}(I_n \otimes E_{jj})Y = 1, \operatorname{tr}(E_{jj} \otimes I_n)Y = 1, \quad j = 1, \dots, n \\ (\text{QAP}_{\text{ZW}}) \quad & \operatorname{tr}(I_n \otimes (J_n - I_n) + (J_n - I_n) \otimes I_n)Y = 0 \\ & \operatorname{tr}(J_{n^2} Y) = n^2 \\ & Y \geq 0, \quad Y \succeq 0. \end{aligned}$$

We use the abbreviation ZW to emphasize that this model is motivated by Zhao et al. [66]. Namely, in [60] it is proven that the relaxation QAP<sub>ZW</sub> is equivalent to the earlier relaxation of Zhao et al. [66]. The SDP relaxation QAP<sub>ZW</sub> is also equivalent to the so-called  $N^+(K)$ -relaxation by Lovász and Schrijver [53] applied to the QAP, as was observed by Burer and Vandenbussche [9]. The equivalence between the two relaxations was also shown in [60].

One may easily verify that QAP<sub>ZW</sub> is indeed a relaxation of the QAP (27.1) by noting that

$$Y := \text{vec}(X)\text{vec}(X)^T$$

is a feasible point of QAP<sub>ZW</sub> for  $X \in \Pi_n$ , and that the objective value of QAP<sub>ZW</sub> at this point  $Y$  is precisely  $\text{tr}(AXB + C)X^T$ . The constraints involving  $E_{jj}$  are generalizations of the assignment constraints  $Xu = X^Tu = u$ , and the sparsity constraints i.e.,  $\text{tr}(I_n \otimes (J_n - I_n) + (J_n - I_n) \otimes I_n)Y = 0$  generalize the orthogonality conditions  $XX^T = X^TX = I$ . This sparsity pattern is sometimes called the *Gangster constraint* and is denoted by  $\mathcal{G}(Y) = 0$ , see e.g., [66].

The following lemma from [60] gives an explicit description of the feasible set of QAP<sub>ZW</sub>.

**Theorem 27.1 ([60], Lemma 6).** *A matrix*

$$Y = \begin{pmatrix} Y^{(11)} & \dots & Y^{(1n)} \\ \vdots & \ddots & \vdots \\ Y^{(n1)} & \dots & Y^{(nn)} \end{pmatrix} \in \mathcal{S}_{n^2}^+, \quad Y^{(ij)} \in \mathbb{R}^{n \times n}, \quad i, j = 1, \dots, n,$$

is feasible for QAP<sub>ZW</sub> if and only if  $Y$  satisfies

- (i)  $\mathcal{G}(Y) = 0$ ,  $\text{tr}(Y^{(ii)}) = 1$  for  $1 \leq i \leq n$ ,  $\sum_{i=1}^n \text{diag}(Y^{(ii)}) = u$ ,
- (ii)  $u^T Y^{(ij)} = \text{diag}(Y^{(jj)})^T$  for  $1 \leq i, j \leq n$ ,
- (iii)  $\sum_{i=1}^n Y^{(ij)} = u \text{ diag}(Y^{(jj)})^T$  for  $1 \leq j \leq n$ .

Bounds obtained by solving QAP<sub>ZW</sub> are quite good in practice, but computationally demanding for interior point solvers, even for relatively small instances (say  $n \geq 15$ ). (Note that the matrix variable  $Y$  is a square matrix of order  $n^2$ , and that the relaxation has  $O(n^4)$  sign constraints and  $O(n^3)$  equality constraints.) Therefore, some alternative methods were used to solve QAP<sub>ZW</sub>. A dynamic version of the bundle method is used in [62] to approximately solve QAPLIB [10] problems of sizes  $n \leq 32$ , and in [9] the augmented Lagrangian approach is implemented for solving problems with  $n \leq 36$ .

For QAP instances where the data matrices have large *automorphism groups*, the SDP bounds can be computed more efficiently by exploiting symmetry. This was shown in [20], see also Chap. 7 of this Handbook. In [20], QAP<sub>ZW</sub> bounds for Eschermann and Wunderlich [28] instances for  $n$  up to 128 are computed with interior point solvers. In these instances the automorphism group of  $B$  is the automorphism group of the Hamming graph (see e.g. [64], or Example 5.1 in [20]).

**Table 27.1** Optimal values for the larger esc instances

Instance	Previous l.b.	$\text{QAP}_{\text{ZW}}$	$\text{QAP}_{\text{dKS}}$	Best known u.b.
esc32a	103 ([9])	104	107	130
esc32h	424 ([9])	425	427	438
esc64a	47	98	105	116
esc128	2	54	n.a.	64

In Table 27.1,  $\text{QAP}_{\text{ZW}}$  bounds are listed for Eschermann–Wunderlich instances of sizes  $n = 32, 64, 128$  that are computed using symmetry reduction (for details see [20]). Note that bounds from [9] do not always give the same bounds as those from [20]. The reason for the difference is that the augmented Lagrangian method does not always solve the SDP relaxation  $\text{QAP}_{\text{ZW}}$  to optimality. The previous lower bounds for esc64a and esc128 given in Table 27.1 are the Gilmore–Lawler [34, 50] bounds.

Thus, in [20] it is shown that QAP instances where the data matrices have large automorphism groups,  $\text{QAP}_{\text{ZW}}$  bounds can be computed efficiently by IPMs. Moreover, this approach gives an opportunity for solving strengthened  $\text{QAP}_{\text{ZW}}$  relaxations.

As noted in [62], the SDP relaxation  $\text{QAP}_{\text{ZW}}$  can be strengthened by adding the so called *triangle inequalities*

$$0 \leq Y_{rs} \leq Y_{rr} \quad (27.2)$$

$$Y_{rr} + Y_{ss} - Y_{rs} \leq 1 \quad (27.3)$$

$$-Y_{tt} - Y_{rs} + Y_{rt} + Y_{st} \leq 0 \quad (27.4)$$

$$Y_{tt} + Y_{rr} + Y_{ss} - Y_{rs} - Y_{rt} - Y_{st} \leq 1, \quad (27.5)$$

which hold for all distinct triples  $(r, s, t)$ . Note that there are  $O(n^6)$  triangle inequalities. In [20] the following result is proven.

**Lemma 27.1.** [20] *If an optimal solution  $Y$  of  $\text{QAP}_{\text{ZW}}$  has constant diagonal, then all the triangle inequalities (27.2)–(27.5) are satisfied.*

Since optimal solutions of  $\text{QAP}_{\text{ZW}}$  for the instances solved in [20] have constant diagonal, the lower bounds can not be improved by adding the triangle inequalities (27.2)–(27.5).

In [21] it is shown how one may obtain stronger bounds than  $\text{QAP}_{\text{ZW}}$  for QAP instances where one of the data matrices has a *transitive automorphism group*. The results in the sequel are restricted to the QAP where  $C = 0$ . The idea in [21] is to assign facility  $s$  to location  $r$  for a given arbitrary index pair  $(r, s)$ . In other words, to fix entry  $(r, s)$  in the permutation matrix  $X$  to one. This leads to a QAP problem that is one dimension smaller than the original one, i.e., to the problem

$$\min_{X \in \Pi_{n-1}} \text{tr}(A(\alpha)XB(\beta) + \bar{C}(\alpha, \beta))X^T. \quad (27.6)$$

Here  $\alpha = \{1, \dots, n\} \setminus r$  and  $\beta = \{1, \dots, n\} \setminus s$ , and

$$\bar{C}(\alpha, \beta) = 2A(\alpha, \{r\})B(\{s\}, \beta).$$

Further, in [21] it is suggested to solve relaxation QAP<sub>ZW</sub> for the corresponding smaller dimensional problem, i.e.,

$$\begin{aligned} & \min \operatorname{tr}(B(\beta) \otimes A(\alpha) + \operatorname{Diag}(\operatorname{vec}(\bar{C})))Y \\ \text{s.t. } & \operatorname{tr}(I_{n-1} \otimes E_{jj})Y = 1, \operatorname{tr}(E_{jj} \otimes I_{n-1})Y = 1, j = 1, \dots, n-1 \\ (\text{QAP}_{\text{dKS}}) \quad & \operatorname{tr}(I_{n-1} \otimes (J_{n-1} - I_{n-1}) + (J_{n-1} - I_{n-1}) \otimes I_{n-1})Y = 0 \\ & \operatorname{tr}(J_{(n-1)^2} Y) = (n-1)^2 \\ & Y \geq 0, Y \succeq 0. \end{aligned}$$

Note that here  $Y$  has size  $(n-1)^2 \times (n-1)^2$ . In [21] the symmetry reduction of the SDP problem QAP<sub>dKS</sub> is performed in order to solve the relaxation efficiently.

In general, the bounds obtained from QAP<sub>dKS</sub> are not lower bounds for the original QAP problem, but if  $\operatorname{aut}(A)$  or  $\operatorname{aut}(B)$  is transitive then they are also global lower bounds.

**Theorem 27.2 ([21]).** *If  $\operatorname{aut}(A)$  or  $\operatorname{aut}(B)$  is transitive and  $C = 0$ , then any lower bound for the QAP subproblem (27.6) at the first level of the branching tree is also a lower bound for the original QAP.*

Note that if both  $\operatorname{aut}(A)$  and  $\operatorname{aut}(B)$  are transitive, then there is only one subproblem. However, if only one of the automorphisms groups of the data matrices is transitive, then the number of different subproblems depends on the automorphism group of the other data matrix. In any case, if only one of the automorphisms groups of the data matrices is transitive, the number of different subproblems is at most  $n$ , the order of the data matrices. The following lemma provides details.

**Lemma 27.2 ([21]).** *Let  $\operatorname{aut}(B)$  be transitive. Then there are as many different child subproblems at the first level of the branching tree as there are orbits of  $\operatorname{aut}(A)$ .*

In the sequel we discuss the quality of the bounds QAP<sub>dKS</sub> obtained by fixing arbitrary  $(r, s)$ . In order to do that, we need to show that the following SDP problem

$$\begin{aligned} & \min \operatorname{tr}(B \otimes A)Y \\ \text{s.t. } & \operatorname{tr}(I_n \otimes E_{jj})Y = 1, \operatorname{tr}(E_{jj} \otimes I_n)Y = 1, \quad j = 1, \dots, n \\ & \operatorname{tr}(I_n \otimes (J_n - I_n) + (J_n - I_n) \otimes I_n)Y = 0 \\ & \operatorname{tr}(J_{n^2} Y) = n^2 \\ & \operatorname{tr}(E_{ss} \otimes E_{rr})Y = 1 \\ & Y \geq 0, Y \succeq 0, \end{aligned} \tag{27.7}$$

is equivalent to QAP<sub>dKS</sub>. Note that SDP relaxation (27.7) differs from QAP<sub>ZW</sub> by one additional constraint. For the proof of equivalence we will use the following lemmas.

**Lemma 27.3.** *Let  $Y$  be feasible for (27.7), then*

- (a)  $\text{tr}(E_{jj} \otimes E_{rr})Y = 0, j = 1, \dots, n, j \neq s$
- (b)  $\text{tr}(E_{ss} \otimes E_{jj})Y = 0, j = 1, \dots, n, j \neq r$
- (c)  $\text{tr}(E_{ij} \otimes (E_{rk} + E_{kr}))Y = 0, i \neq j, k = 1, \dots, n$
- (d)  $\text{tr}((E_{sk} + E_{ks}) \otimes E_{ij})Y = 0, k \neq s, i \neq r, j \neq r$ .

*Proof.* The results follow trivially from the generalized assignment constraints in (27.7), and from the fact that if  $Y \in \mathcal{S}_n^+$  and  $y_{ii} = 0$ , then  $y_{ij} = 0$ , for all  $j = 1, \dots, n$ .  $\square$

**Lemma 27.4.** *Let  $Y$  be feasible for (27.7), then*

$$\text{diag}(Y) = Y_{:(s-1)n+r}.$$

*Proof.* This follows from the fact that  $u_n^T Y^{(sj)} = \text{diag}(Y^{(sj)})^T$  for  $1 \leq j \leq n$  (see Theorem 27.1) and (c)–(d) in Lemma 27.3.  $\square$

Now we can show that (27.7) and QAP<sub>dKS</sub> are equivalent. A similar proof was obtained by De Klerk (private communication, 2010).

**Theorem 27.3.** *Let  $r, s \in \{1, \dots, n\}$  be fixed. The semidefinite program (27.7) is equivalent to QAP<sub>dKS</sub> in the sense that there is a bijection between the feasible sets that preserves the objective function.*

*Proof.* Assume w.l.o.g. that  $r = s = 1$ . We show first that for any  $Y \in \mathcal{S}_{n^2 \times n^2}$  that is feasible for (27.7), we can find exactly one matrix  $Z = Z(Y) \in \mathcal{S}_{(n-1)^2 \times (n-1)^2}$  that is feasible for QAP<sub>dKS</sub>. Consider the block form of  $Y$  as in Theorem 27.1. From Lemma 27.3 and Lemma 27.4 it follows that

$$Y_{rk}^{(ij)} = Y_{kr}^{(ij)} = 0 \quad i, j = 2, \dots, n, k = 1, \dots, n,$$

and for  $\alpha = \{2, \dots, n\}$  the following principal submatrices are zero matrices, i.e.,

$$Y^{(1j)}(\alpha) = Y^{(j1)}(\alpha) = 0_{n-1}, \quad j = 1, \dots, n,$$

and

$$Y_{:,1}^{(j1)} = \text{diag}(Y^{(jj)}), \quad j = 1, \dots, n.$$

Now, we define

$$Z^{(i-1,j-1)} := Y^{(ij)}(\alpha) \in \mathbb{R}^{(n-1) \times (n-1)}, \quad i, j = 2, \dots, n,$$

and

$$Z(Y) := \begin{pmatrix} Z^{(11)} & \dots & Z^{(1,n-1)} \\ \vdots & \ddots & \vdots \\ Z^{(n-1,1)} & \dots & Z^{(n-1,n-1)} \end{pmatrix} \in \mathcal{S}_{(n-1)^2 \times (n-1)^2}.$$

Since  $Z(Y)$  is a principal submatrix of the positive semidefinite matrix  $Y$ , it follows that  $Z(Y) \geq 0$ . The other constraints from QAP<sub>dKS</sub> are trivially satisfied.

Conversely, let  $Z \in \mathcal{S}_{(n-1)^2 \times (n-1)^2}^+$  be feasible for QAP<sub>dKS</sub>. We construct  $Y \in \mathcal{S}_{n^2 \times n^2}^+$  such that  $Z = Z(Y)$  in the following way. In each block  $Z^{(ij)}$ ,  $i, j = 1, \dots, n-1$ , add the first row and column with all zeros and call these  $n \times n$  blocks  $\bar{Z}^{(ij)}$ . Define  $n$  blocks  $\bar{Z}^{(1j)} \in \mathbb{R}^{n \times n}$ ,  $j = 1, \dots, n$  in the following way

$$\bar{Z}_{ij}^{(11)} = \begin{cases} 1 & \text{for } i = j = 1 \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\bar{Z}_{1,:}^{(1j)} = \text{diag}(\bar{Z}^{(jj)})^T, \quad \bar{Z}_{kl}^{(1j)} = 0, \quad l = 1, \dots, n, k, j = 2, \dots, n.$$

Finally, let

$$Y := \begin{pmatrix} \bar{Z}^{(11)} & \bar{Z}^{(12)} & \dots & \bar{Z}^{(1n)} \\ (\bar{Z}^{(12)})^T & \bar{Z}^{(11)} & \dots & \bar{Z}^{(1,n-1)} \\ \vdots & \ddots & \vdots & \vdots \\ (\bar{Z}^{(1n)})^T & \bar{Z}^{(n1)} & \dots & \bar{Z}^{(n-1,n-1)} \end{pmatrix} \in \mathcal{S}_{n^2 \times n^2}.$$

From Proposition 27.1 it follows that  $Y$  is a positive semidefinite matrix. Also the remaining constraints in (27.7) are satisfied, due to the construction of  $Y$ .

Finally, note that for any pair of feasible solutions  $Y \in \mathcal{S}_{n^2 \times n^2}^+$  of (27.7) and  $Z = Z(Y) \in \mathcal{S}_{(n-1)^2 \times (n-1)^2}^+$  of QAP<sub>dKS</sub> it holds that

$$\text{tr}(B \otimes A)Y = \text{tr}(B(\beta) \otimes A(\alpha) + 2\text{Diag}(B(\beta, \{s\}) \otimes A(\alpha, \{r\})))Z.$$

□

Finally, we may conclude.

**Corollary 27.1.** *Fix  $r, s \in \{1, \dots, n\}$ . Assume that  $\text{aut}(A)$  or  $\text{aut}(B)$  is transitive and  $C = 0$ . Then the SDP relaxation QAP<sub>dKS</sub> dominates QAP<sub>ZW</sub>.*

The numerical results in [21] (see also Table 27.1) show that the SDP lower bounds QAP<sub>dKS</sub> can be significantly better than the SDP bounds QAP<sub>ZW</sub>. The authors in [21] were not able to form QAP<sub>dKS</sub> relaxation for esc128 problem due to the memory issues.

The SDP bound QAP<sub>dKS</sub> has several potential areas of application. In particular, it is applicable for all QAP instances where the automorphism group of one of the data matrices is transitive. Famous examples of such instances are the traveling

salesman problem and the graph equipartition problem, which will be discussed in the next sections.

### 27.3 The Traveling Salesman Problem

Let  $K_n(D)$  be a complete graph on  $n$  vertices with edge lengths  $D_{ij} = D_{ji} > 0$  ( $i \neq j$ ). The traveling salesman problem (TSP) is to find a Hamiltonian circuit of minimum length in  $K_n(D)$ . The  $n$  vertices are often called cities, and the Hamiltonian circuit of minimum length the optimal tour. Here we present two SDP relaxations for the traveling salesman problem.

Cvetković et al. [14] derived a semidefinite programming relaxation for the TSP that is motivated by the algebraic connectivity of graphs. (The algebraic connectivity of a graph is defined as the second smallest eigenvalue of the Laplacian of the graph, see (27.8).) In particular, Cvetković et al. [14] exploit the fact that the Laplacian eigenvalues of the Hamiltonian circuit are  $2(1 - \cos(\frac{2k\pi}{n}))$  for  $k = 0, \dots, n-1$ , and the second smallest eigenvalue is

$$h_n := 2 \left( 1 - \cos \left( \frac{2\pi}{n} \right) \right).$$

In [14] it is concluded that if  $X$  is in the convex hull of all Hamiltonian cycles (represented by their adjacency matrices) through  $n$  vertices, then its algebraic connectivity is at least  $h_n$ . This eigenvalue constraint is reformulated as a semidefinite constraint, and the resulting SDP relaxation for the TSP from [14] is:

$$\begin{aligned} & \min \frac{1}{2} \operatorname{tr}(DX) \\ \text{s.t. } & Xu_n = 2u_n \\ (\text{TSP}_{\text{CK}}) \quad & \operatorname{diag}(X) = 0 \\ & 0 \leq X \leq J \\ & 2I - X + 2 \left( 1 - \cos \left( \frac{2\pi}{n} \right) \right) (J - I) \geq 0. \end{aligned}$$

It was shown by Goemans and Rendl [35] that the well known Held–Karp linear programming bound [17, 39] dominates the SDP relaxation  $\text{TSP}_{\text{CK}}$ . In the sequel we present the most recent SDP bound for the TSP. This bound dominates  $\text{TSP}_{\text{CK}}$  and is independent of the Held–Karp bound.

In [19], a new SDP relaxation of the TSP is derived. This relaxation coincides with the SDP relaxation for the QAP introduced in [66] when applied to the QAP reformulation of the TSP. Indeed, it is well known that the TSP is a special case of the QAP, and can be stated as

$$(\text{TSP}) \quad \min_{X \in \mathcal{P}_n} \operatorname{tr} \left( \frac{1}{2} D X C_n X^T \right),$$

where  $C_n$  is the adjacency matrix of the canonical circuit on  $n$  vertices:

$$C_n := \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & & & 0 & 1 \\ 1 & 0 & \cdots & 0 & 1 & 0 \end{pmatrix},$$

and  $D$  is the matrix of edge lengths. Thus, the following SDP relaxation of the TSP is derived in [19] from QAP<sub>ZW</sub> (see p. 797) by symmetry reduction, see also Chap. 7 of this Handbook.

$$\begin{aligned} & \min \frac{1}{2} \operatorname{tr}(DX^{(1)}) \\ & \text{s.t. } \sum_{k=1}^d X^{(k)} = J - I \\ (\text{TSP}_{\text{dKPS}}) \quad & I + \sum_{k=1}^d \cos\left(\frac{2\pi ik}{n}\right) X^{(k)} \geq 0 \quad i = 1, \dots, d \\ & X^{(k)} \geq 0, \quad X^{(k)} \in \mathcal{S}_n, \quad k = 1, \dots, d, \end{aligned}$$

where  $d = \lfloor \frac{1}{2}n \rfloor$  is the diameter of the canonical circuit on  $n$  vertices. In [19] it is proven that TSP<sub>dKPS</sub> dominates the relaxation due to Cvetković et al. [14], and is not dominated by the Held–Karp linear programming bound (or vice versa). In [22] it is shown that when the matrix of distances between cities  $D$  is symmetric and circulant, the SDP relaxation TSP<sub>dKPS</sub> reduces to a linear programming problem.

Following the idea on improving QAP<sub>ZW</sub> by fixing one position in the permutation matrices, one can strengthen TSP<sub>dKPS</sub> as well. This approach is applicable to the TSP since the automorphism group of  $C_n$  is the dihedral group, which is transitive. Preliminary results in [21] show that the lower bounds so obtained are very promising. Finally, we note here that the SDP bound TSP<sub>dKPS</sub> is hard to compute for large size problems; however it provides a new polynomial-time convex approximation of the TSP with a rich mathematical structure.

## 27.4 The Graph Equipartition Problem

The  $k$ -equipartition problem ( $k$ -GP) is defined as follows. Consider an undirected graph  $G = (V, E)$  with vertex set  $V$ ,  $|V| = n = km$ , and edge set  $E$ . The goal is to find a partition of the vertex set into  $k$  sets  $S_1, \dots, S_k$  of equal cardinality, i.e.,  $|S_i| = m$ ,

$i = 1, \dots, k$  such that the total weight of edges joining different sets  $S_i$  is minimized. If there is no restriction on the size of the sets, then we refer to this as the *graph partition* problem.

We denote by  $A$  the *adjacency matrix* of  $G$ . For a given partition of the graph into  $k$  subsets, let  $X = (x_{ij})$  be the  $n \times k$  matrix defined by

$$x_{ij} = \begin{cases} 1 & \text{if node } i \text{ is in } S_j \\ 0 & \text{if node } i \text{ is not in } S_j. \end{cases}$$

It is clear that the  $j$ th column  $X_{:,j}$  is the characteristic vector of  $S_j$ , and  $k$ -partitions are in one-to-one correspondence with the set

$$\mathcal{P}_k := \left\{ X \in \mathbb{R}^{n \times k} : Xu_k = u_n, X^T u_n = mu_k, x_{ij} \in \{0, 1\} \right\}.$$

For each  $X \in \mathcal{P}_k$ , it holds that:

- $\text{tr}X^TDX = \text{tr}D$ , if  $D$  is diagonal.
- $\frac{1}{2} \text{tr}X^TAX$  gives the total weight of the edges within the subsets  $S_j$ .

Therefore, the total weight of edges cut by  $X$ , i.e., those joining different sets  $S_j$  is

$$w(E_{\text{cut}}) := \frac{1}{2} \text{tr}(X^T LX),$$

where

$$L := \text{Diag}(Au_n) - A \quad (27.8)$$

is the Laplacian matrix of the graph. The  *$k$ -equipartition problem* in a trace formulation can then be written:

$$\begin{array}{ll} \min & \frac{1}{2} \text{tr}(X^T LX) \\ (\text{k-GP}) & \text{s.t. } X \in \mathcal{P}_k \end{array}$$

It is easy to verify that for  $X \in \mathcal{P}_k$  it follows:

$$\text{tr}(X^T LX) = \text{tr}(X^T AX(J_k - I_k)), \quad (27.9)$$

where  $J_k$  is the  $k \times k$  all-ones matrix and  $I_k$  the identity matrix of order  $k$ .

The graph partition problem has many applications such as VLSI design [51], parallel computing [6] and floor planing [16]. The equipartition problem also plays a role in telecommunications, see e.g., [52]. For more applications of the  $k$ -equipartition problem see Lengauer [43] and the references therein.

The  $k$ -equipartition problem is NP-hard, even for  $k = 2$ , see [31]. The graph equipartition problem is extensively studied and many heuristics are suggested,

see e.g., Kernighan and Lin [46], Fiduccia and Mattheyses [29]. In [32], a branch-and-cut algorithm based on SDP is implemented for the minimum  $k$ -partition problem. The authors here compute globally optimal solutions for dense graphs up to 60 vertices, for grid graphs with up to 100 vertices, and for different values of  $k$ . Also, there are several known relaxations of the problem, and we list some of them below. In 1973, Donath and Hoffman [25] derive an eigenvalue based bound for  $k$ -GP that was further improved by Rendl and Wolkowicz [63] in 1995. In [1], Alizadeh shows that the Donath–Hoffman bound can be obtained as the dual of a semidefinite program. A detailed description of these bounds and their relationships are given by Karisch and Rendl [45]. In the same paper bounds are derived that are stronger than the mentioned eigenvalue based bounds, whereas Wolkowicz and Zhao [67] derive bounds for the general graph partition problem.

In the sequel we derive an SDP relaxation from [45], which is based on matrix lifting, and an SDP relaxation from [67], which is based on vector lifting, and show that they are *equivalent*. Moreover, we relate these relaxations with an SDP relaxation of  $k$ -equipartition problem that is derived from the SDP relaxation of the quadratic assignment problem. In particular, we first derive the SDP relaxation  $k$ -GP<sub>KR</sub> [45] whose matrix variable is in  $\mathcal{S}_n^+$ . Then, we derive the SDP relaxation  $k$ -GP<sub>ZW</sub> [67] that also includes nonnegativity constraints and whose matrix variable is in  $\mathcal{S}_{nk+1}^+$ . Furthermore, we introduce a new relaxation  $k$ -GP<sub>RS</sub> with matrix variable in  $\mathcal{S}_{nk}^+$  that we prove is dominated by  $k$ -GP<sub>ZW</sub> and which dominates  $k$ -GP<sub>KR</sub>. Finally, we introduce an SDP relaxation for the graph equipartition problem that is derived as a special case of the QAP in [23] and that is known to be equivalent to  $k$ -GP<sub>KR</sub>, and show that this relaxation dominates  $k$ -GP<sub>ZW</sub>. From this follows the equivalence of all these formulations.

Very recently, a preprint [26] has appeared in which it is also shown that SDP relaxations based on vector-lifting and on matrix lifting for quadratically constrained quadratic programs with matrix variables provide equivalent bounds, under mild assumptions.

At the end of this section, we present a relaxation for the  $k$ -equipartition problem that was recently derived by De Klerk et al. [23] and that dominates all the above-mentioned relaxations.

One way to obtain a relaxation of the  $k$ -equipartition is to linearize the objective function  $\text{tr}(X^T L X) = \text{tr}(L X X^T)$  by replacing  $XX^T$  by a new variable  $Y$ . This yields the following feasible set of  $k$ -GP:

$$P_k := \text{conv}\left\{Y : \exists X \in \mathcal{P}_k \text{ such that } Y = XX^T\right\}.$$

In order to obtain tractable relaxations for  $k$ -GP one needs to approximate the set  $P_k$  by larger sets containing  $P_k$ . Karisch and Rendl [45] impose on elements  $Y \in P_k$  the following set of constraints

$$\{Y \in \mathcal{S}_n : \text{diag}(Y) = u_n, Y u_n = m u_n, Y \geq 0, Y \succeq 0\},$$

and consequently obtain the SDP relaxation:

$$\begin{aligned} \min & \frac{1}{2} \operatorname{tr}(LY) \\ (\text{k-GP}_{\text{KR}}) \quad \text{s.t. } & \operatorname{diag}(Y) = u_n, Yu_n = mu_n \\ & Y \geq 0, \quad Y \geq 0. \end{aligned}$$

The nonnegativity constraints in k-GP<sub>KR</sub> are redundant when  $k = 2$ , see [45]. In [45] it is further suggested to add to k-GP<sub>KR</sub> the *independent set constraints*

$$\sum_{i < j, i, j \in I} y_{ij} \geq 1, \text{ for all } I \text{ with } |I| = k + 1, \quad (27.10)$$

and the *triangle constraints*

$$y_{ij} + y_{ik} \leq 1 + y_{jk} \text{ for all triples } (i, j, k). \quad (27.11)$$

Adding these two sets of constraints to k-GP<sub>KR</sub> results in a model that contains  $\binom{n}{k+1}$  additional inequalities from (27.10) and  $3\binom{n}{3}$  from (27.11). The resulting relaxation is currently the strongest known relaxation for the  $k$ -equipartition problem. However, it is computationally demanding to obtain the resulting bounds due to the large number of constraints.

Another approach to obtain tractable relaxations of k-GP is to linearize the objective function by lifting into the space of  $(nk + 1) \times (nk + 1)$  matrices as described in [67]. In [67], Wolkowicz and Zhao derive an SDP relaxation for the general graph partition problem. Here, we restrict their approach to the equipartition problem.

For  $X \in \mathcal{P}_k$  we define  $y := \operatorname{vec}(X)$  and  $Y := yy^T$ . We write  $Y$  in block form

$$Y = \begin{pmatrix} Y^{(11)} & \dots & Y^{(1k)} \\ \vdots & \ddots & \vdots \\ Y^{(k1)} & \dots & Y^{(kk)} \end{pmatrix}, \quad (27.12)$$

where

$$Y^{(ij)} := X_{:,i}X_{:,j}^T \in \mathbb{R}^{n \times n}, \quad i, j = 1, \dots, k.$$

Finally, we associate  $X \in \mathcal{P}_k$  with a rank-one matrix  $Y_X \in \mathcal{S}_{nk+1}^+$  in the following way:

$$Y_X := \begin{pmatrix} 1 \\ \operatorname{vec}(X) \end{pmatrix} \begin{pmatrix} 1 \\ \operatorname{vec}(X) \end{pmatrix}^T = \begin{pmatrix} 1 & y^T \\ y & Y \end{pmatrix}, \quad (27.13)$$

which has block form

$$Y_X = \begin{pmatrix} 1 & (y^{(1)})^T & \dots & (y^{(k)})^T \\ y^{(1)} & Y^{(11)} & \dots & Y^{(1k)} \\ \vdots & \vdots & \ddots & \vdots \\ y^{(k)} & Y^{(k1)} & \dots & Y^{(kk)} \end{pmatrix}, \quad (27.14)$$

where  $y^{(i)} = X_{:,i}$ ,  $i = 1, \dots, k$ . The matrix  $Y_X$  has the *sparsity pattern*

$$\text{tr}(E_{ll}Y^{(ij)}) = 0, \quad \forall i, j = 1, \dots, k, \quad i \neq j, \quad l = 1, \dots, n,$$

and any  $Y \in S_{nk}$ ,  $Y \geq 0$  has the same sparsity pattern if and only if

$$\text{tr}((J_k - I_k) \otimes I_n)Y = 0. \quad (27.15)$$

This sparsity pattern, which is similar to that described for QAP<sub>ZW</sub> in Sect. 27.2, is sometimes called the Gangster constraint, see [67].

The constraints  $Xu_k = u_n$  and  $X^T u_n = mu_k$  are equivalent to

$$T \begin{pmatrix} 1 \\ \text{vec}(X) \end{pmatrix} = 0, \quad (27.16)$$

where

$$T := \begin{pmatrix} -mu_k & I_k \otimes u_n^T \\ -u_n & u_k^T \otimes I_n \end{pmatrix}.$$

Constraint (27.16) may be rewritten as

$$\text{tr}(T^T T Y_X) = 0,$$

where

$$T^T T = \begin{pmatrix} km^2 + n & -(m+1)u_{nk}^T \\ -(m+1)u_{nk} & I_k \otimes J_n + J_k \otimes I_n \end{pmatrix}.$$

For any  $Y \geq 0$  that satisfies (27.15), one has

$$\text{tr}\left(T^T T \begin{pmatrix} 1 & y^T \\ y & Y \end{pmatrix}\right) = (km^2 + n) - 2(m+1)u_{nk}^T y + \text{tr}(I_k \otimes J_n)Y + \text{tr}(Y).$$

Thus the condition

$$\text{tr}\left(T^T T \begin{pmatrix} 1 & y^T \\ y & Y \end{pmatrix}\right) = 0 \quad (27.17)$$

becomes

$$\text{tr}(I_k \otimes J_n)Y + \text{tr}(Y) = (m+1)(2u_{nk}^T y - n).$$

Finally, by relaxing the rank-one condition on  $Y_X$  (see (27.13)) to  $Y_X \in \mathcal{S}_{nk+1}^+$  we obtain an SDP relaxation of the  $k$ -equipartition problem (see also [67]):

$$(k\text{-GP}_{ZW}) \quad \begin{aligned} & \min \frac{1}{2} \operatorname{tr}(I_k \otimes L) Y \\ & \text{s.t. } \operatorname{tr}((J_k - I_k) \otimes I_n) Y = 0 \\ & \quad \operatorname{tr}(I_k \otimes J_n) Y + \operatorname{tr}(Y) = (m+1)(2u_{nk}^T y - n) \\ & \quad \begin{pmatrix} 1 & y^T \\ y & Y \end{pmatrix} \in \mathcal{S}_{nk+1}^+, \quad Y \geq 0. \end{aligned}$$

In [67], the nonnegativity constraints are not imposed in the relaxation. Although Wolkowicz and Zhao [67] do not include these  $O(n^4)$  inequalities, they conclude that it would be worth adding them. The following theorem lists the valid constraints that are implied by the constraints of the SDP problem  $k$ -GP<sub>ZW</sub>.

**Theorem 27.4 ([67], Lemma 4.1).** *Assume that  $Y \in \mathcal{S}_{nk}$  and  $y \in \mathbb{R}^{nk}$  are such that*

$$\bar{Y} := \begin{pmatrix} 1 & y^T \\ y & Y \end{pmatrix} \geq 0,$$

*and  $\bar{Y}$  has the block form (27.14) and satisfies (27.17). Then*

- (i)  $\sum_{i=1}^k y^{(i)} = u_n$ ,  $u_n^T y^{(i)} = m$ ,  $i = 1, \dots, k$ .
- (ii)  $m(y^{(j)})^T = u_n^T Y^{(ij)}$ ,  $i, j = 1, \dots, k$ .
- (iii)  $\sum_{i=1}^k Y^{(ij)} = u_n (y^{(j)})^T$ ,  $j = 1, \dots, k$ .
- (iv)  $\sum_{i=1}^k \operatorname{diag}(Y^{ij}) = y^{(j)}$ ,  $j = 1, \dots, k$ .

*Proof.* The proof follows from the fact that  $T\bar{Y} = 0$ . □

The following corollary is an immediate consequence of part (iv) of the previous theorem.

**Corollary 27.2.** *Assume that  $Y \in \mathcal{S}_{nk}$  and  $y \in \mathbb{R}^{nk}$  are as in Theorem 27.4. If  $Y \geq 0$  also satisfies (27.15), then  $\operatorname{diag}(Y) = y$ .*

We now introduce the SDP relaxation  $k$ -GP<sub>RS</sub> as an intermediate step to show the equivalence of  $k$ -GP<sub>ZW</sub> and  $k$ -GP<sub>KR</sub>. In particular, we first show that the following SDP problem is dominated by  $k$ -GP<sub>ZW</sub>:

$$(k\text{-GP}_{RS}) \quad \begin{aligned} & \min \frac{1}{2} \operatorname{tr}(I_k \otimes L) Y \\ & \text{s.t. } \operatorname{tr}((J_k - I_k) \otimes I_n) Y = 0 \\ & \quad \operatorname{tr}(I_k \otimes E_{jj}) Y = 1, \quad j = 1, \dots, n \\ & \quad m \operatorname{diag}(Y^{(ii)}) = Y^{(ii)} u_n, \quad i = 1, \dots, k \\ & \quad \operatorname{tr}(I_k \otimes J_n) Y = km^2 \\ & \quad \operatorname{tr}((J_k - I_k) \otimes J_n) Y = n(n-m) \\ & \quad Y \in \mathcal{S}_{nk}^+, \quad Y \geq 0, \end{aligned}$$

where  $Y$  has block form (27.12). In the following two theorems we compare the relaxations  $k\text{-GP}_{ZW}$ ,  $k\text{-GP}_{RS}$  and  $k\text{-GP}_{KR}$ .

**Theorem 27.5.** *The SDP relaxation  $k\text{-GP}_{ZW}$  dominates the SDP relaxation  $k\text{-GP}_{RS}$ .*

*Proof.* Because the objective functions are the same, it suffices to show that if  $(Y, y)$  is feasible for  $k\text{-GP}_{ZW}$ , then  $Y$  is feasible for  $k\text{-GP}_{RS}$ . Assume that  $Y$  has block form (27.12).

It is clear that  $Y \geq 0$ ,  $Y \geq 0$  and  $\text{tr}((J_k - I_k) \otimes I_n)Y = 0$ . From Theorem 27.4 it follows that  $\text{tr}(I_k \otimes E_{jj})Y = 1$ ,  $j = 1, \dots, n$ , and  $m \text{ diag}(Y^{(ii)}) = Y^{(ii)}u_n$ ,  $i = 1, \dots, k$ . Furthermore, from  $u_n^T Y^{(jj)} = m(\text{diag}(Y^{(jj)}))^T$ ,  $j = 1, \dots, k$ , we have  $u_n^T Y^{(jj)} u_n = m^2$ , and hence constraint  $\text{tr}(I_k \otimes J_n)Y = km^2$  is satisfied. Similarly  $\text{tr}((J_k - I_k) \otimes J_n)Y = n(n - m)$  follows.  $\square$

**Theorem 27.6.** *The SDP relaxation  $k\text{-GP}_{RS}$  dominates the SDP relaxation  $k\text{-GP}_{KR}$ .*

*Proof.* Let  $Y \in \mathcal{S}_{nk}$  be feasible for  $k\text{-GP}_{RS}$  with block form (27.12). We construct from  $Y$  a feasible point  $\tilde{Y} \in \mathcal{S}_n$  for  $k\text{-GP}_{KR}$  in the following way:

$$\tilde{Y} := \sum_{j=1}^k Y^{(jj)}.$$

Clearly,  $\tilde{Y} \geq 0$  and  $\tilde{Y} \geq 0$ . From  $\text{tr}(I_k \otimes E_{jj})Y = 1$ ,  $j = 1, \dots, n$ , it follows that

$$\text{diag}(\tilde{Y}) = \text{diag}\left(\sum_{j=1}^k Y^{(jj)}\right) = u_n,$$

and from  $Y^{(ii)}u_n = m \text{ diag}(Y^{(ii)})$ ,  $i = 1, \dots, k$ , that

$$\tilde{Y}u_n = \sum_{j=1}^k Y^{(jj)}u_n = m \sum_{j=1}^k \text{diag}(Y^{(jj)}) = mu_n.$$

It remains to show that the objectives agree. Indeed,

$$\text{tr}(I_k \otimes L)Y = \sum_{j=1}^k \text{tr}(LY^{(jj)}) = \text{tr}(L\tilde{Y}).$$

$\square$

Finally, to show equivalence of the SDP relaxations  $k\text{-GP}_{KR}$  and  $k\text{-GP}_{ZW}$  we introduce one more relaxation of the  $k$ -equipartition problem. That relaxation is related to the quadratic assignment problem. Namely, it is well known that the

graph equipartition problem is a special case of the QAP. To show this, we recall that the sets  $\Pi_n$  and  $\mathcal{P}_k$  are related in the following way (see e.g., [44]). If  $Z \in \Pi_n$  then  $X = Z(I_k \otimes u_m) \in \mathcal{P}_k$ . Conversely, each  $X \in \mathcal{P}_k$  can be written as  $X = Z(I_k \otimes u_m)$  with  $Z \in \Pi_n$ . For such a related pair  $(Z, X)$  it follows that

$$\text{tr}(X^T AX(J_k - I_k)) = \text{tr}(Z^T AZ(I_k \otimes u_m)(J_k - I_k)(I_k \otimes u_m)^T) = \text{tr}(Z^T AZB),$$

where

$$B := (J_k - I_k) \otimes J_m \in \mathcal{S}_n. \quad (27.18)$$

Therefore, the  $k$ -equipartition problem may be formulated as the QAP

$$\min_{Z \in \Pi_n} \frac{1}{2} \text{tr}AZBZ^T,$$

where  $A$  is the adjacency matrix of  $G$ , and  $B$  is of the form (27.18). In [23] it is suggested to use the SDP relaxation QAP<sub>ZW</sub> (see p. 797) of the quadratic assignment problem as an SDP relaxation of the  $k$ -equipartition problem. Here we call that relaxation k-GP<sub>QAP</sub>:

$$\begin{aligned} & \min \quad \frac{1}{2} \text{tr}(B \otimes A)Y \\ & \text{s.t. } \text{tr}(I \otimes E_{jj})Y = 1, \quad \text{tr}(E_{jj} \otimes I)Y = 1 \quad j = 1, \dots, n \\ (\text{k-GP}_{\text{QAP}}) \quad & \text{tr}(I \otimes (J - I)) + (J - I) \otimes I)Y = 0 \\ & \text{tr}(J_{n^2}Y) = n^2 \\ & Y \geq 0, \quad Y \geq 0. \end{aligned}$$

This relaxation can be reduced by using symmetry reduction, as is shown by Dobre [24]. By using the reduced formulation of k-GP<sub>QAP</sub>, he proves the following result.

**Theorem 27.7** ([24]). *The SDP problems k-GP<sub>KR</sub> and k-GP<sub>QAP</sub> are equivalent.*

In the following theorem we relate k-GP<sub>QAP</sub> and k-GP<sub>ZW</sub>. To prove that k-GP<sub>ZW</sub> is dominated by k-GP<sub>QAP</sub>, we need the following proposition.

**Proposition 27.1** ([33], Proposition 7). *Let  $A \in \mathcal{S}_n$  such that  $\text{diag}(A) = cAu_n$  for some  $c \in \mathbb{R}$ , and*

$$\bar{A} = \begin{pmatrix} 1 & a^T \\ a & A \end{pmatrix}$$

where  $a := \text{diag}(A)$ . Then the following are equivalent:

- (i)  $\bar{A}$  is positive semidefinite,
- (ii)  $A$  is positive semidefinite and  $\text{tr}(J_n A) \geq (\text{tr}A)^2$ .

**Theorem 27.8.** *The SDP relaxation k-GP<sub>QAP</sub> dominates the SDP relaxation k-GP<sub>ZW</sub>.*

*Proof.* Let  $Y \in \mathcal{S}_{n^2}^+$  be feasible for k-GPQAP with block form (27.12) where  $k = n$  and  $Y^{(ij)} \in \mathbb{R}^{n \times n}$ ,  $i, j = 1, \dots, n$ . We construct from  $Y \in \mathcal{S}_{n^2}^+$  a feasible point  $(W, w)$  with  $W \in \mathcal{S}_{nk}$  for k-GPZW in the following way. First, define blocks

$$W^{(pq)} := \sum_{i=(p-1)m+1}^{pm} \sum_{j=(q-1)m+1}^{qm} Y^{(ij)}, \quad p, q = 1, \dots, k, \quad (27.19)$$

and then collect all  $k^2$  blocks into the matrix:

$$W = \begin{pmatrix} W^{(11)} & \dots & W^{(1k)} \\ \vdots & \ddots & \vdots \\ W^{(k1)} & \dots & W^{(kk)} \end{pmatrix}, \quad (27.20)$$

and set  $w = \text{diag}(W)$ . The sparsity pattern  $\text{tr}((J_k - I_k) \otimes I_n)W = 0$  follows from the sparsity pattern of  $Y$ , i.e., from  $\text{tr}((J_n - I_n) \otimes I_n)Y = 0$ . By direct verification it follows that

$$\text{tr}(I_k \otimes J_n)W + \text{tr}(W) = (m+1)(2u_{nk}^T w - n).$$

To prove that

$$\begin{pmatrix} 1 & w^T \\ w & W \end{pmatrix} \geq 0, \quad (27.21)$$

we use Proposition 27.1. From the valid equalities for k-GPQAP (see Theorem 27.1) it follows that

$$W^{(pp)} u_n = m \text{ diag}(W^{(pp)}) \text{ and } W^{(pq)} u_n = m \text{ diag}(W^{(pp)}), \quad p, q = 1, \dots, k.$$

Therefore, the assumptions of Proposition 27.1 are satisfied. Further, from

$$u_{nk}^T W u_{nk} = n^2 \quad \text{and} \quad \text{tr}(W) = n,$$

we have that  $\text{tr}(J_n W) \geq (\text{tr}W)^2$ . Finally, for any  $x \in \mathbb{R}^{nk}$  let  $\tilde{x} \in \mathbb{R}^{n^2}$  be defined by

$$\tilde{x}^T := [u_m^T \otimes x_{1:n}^T, \dots, u_m^T \otimes x_{n(k-1)+1:nk}^T]$$

then

$$x^T W x = \tilde{x}^T Y \tilde{x} \geq 0,$$

since  $Y \succeq 0$ . Now (27.21) follows from Proposition 27.1.

It remains to show that for any pair of feasible solutions  $(Y, (W, w))$ , which are related as described, the objective values coincide. From (27.19) and (27.9) we have

$$\text{tr}((J_k - I_k) \otimes (J_m \otimes A))Y = \text{tr}((J_k - I_k) \otimes A)W,$$

which finishes the proof.  $\square$

We have now shown the following sequence:

$$k\text{-GP}_{ZW} \geq k\text{-GP}_{RS} \geq k\text{-GP}_{KR} \equiv k\text{-GP}_{QAP} \geq k\text{-GP}_{ZW},$$

where  $A \geq B$  means that  $A$  dominates  $B$ . Hence, we can finally conclude that all formulations are equivalent.

In [23], the authors derive an SDP relaxation for the  $k$ -equipartition problem that dominates the previously described relaxations. That relaxation coincides with the relaxation for the quadratic assignment problem  $\text{QAP}_{dKS}$  (see p. 800) where  $B$  is as in (27.18) and  $A$  is the adjacency matrix of the graph. After applying symmetry reduction to  $\text{QAP}_{dKS}$  the relaxation from [23] takes the form

$$\begin{aligned} & \min a^T \text{diag}(X_5) + \frac{1}{2} \text{tr} \bar{A}(X_3 + X_4 + X_7) \\ & \text{s.t.} \\ & \quad X_1 + X_5 = I_{n-1} \\ & \quad \sum_{t=1}^7 \text{tr}(JX_t) = (n-1)^2 \\ & \quad \text{tr}(X_1) = m-1 \\ & \quad \text{tr}(X_5) = (k-1)m \\ & \quad \text{tr}(X_2 + X_3 + X_4 + X_6 + X_7) = 0 \\ & \quad X_3 = X_4^T \\ & (k\text{-GP}_{SYM}) \left( \begin{array}{cc} \frac{1}{m-1}(X_1 + X_2) & \frac{1}{\sqrt{(k-1)m(m-1)}} X_3 \\ \frac{1}{\sqrt{(k-1)m(m-1)}} X_4 & \frac{1}{(k-1)m}(X_5 + X_6 + X_7) \end{array} \right) \geq 0 \\ & \quad X_1 - \frac{1}{m-2} X_2 \geq 0 \\ & \quad X_5 - \frac{1}{m-1} X_6 \geq 0 \\ & \quad X_5 + X_6 - \frac{1}{k-2} X_7 \geq 0 \\ & \quad X_i \geq 0, i = 1, \dots, 7, \end{aligned}$$

where the  $X_i$  are all of order  $n-1$ ,  $a = A_{:,j}$ , and  $\bar{A} = A(\{1, \dots, j-1, j+1, \dots, n\})$  is the principal submatrix of  $A$ , for some (fixed)  $j = 1, \dots, n$ . The relaxation  $k\text{-GP}_{SYM}$  gives a lower bound for the original problem  $k\text{-GP}$  since the automorphism group of  $B$  is transitive, see Theorem 27.2.

In [23] it is shown that the SDP relaxation  $k\text{-GP}_{SYM}$  dominates  $k\text{-GP}_{KR}$ , and consequently all of the equivalent bounds presented here. However,  $k\text{-GP}_{SYM}$  does not dominate bounds obtained by adding constraints (27.10) and (27.11) to  $k\text{-GP}_{KR}$ .

## 27.5 The Bandwidth Problem in Graphs

Let  $G = (V, E)$  be an undirected graph with  $|V| = n$  vertices and edge set  $E$ . A bijection  $\phi : V = \{v_1, \dots, v_n\} \rightarrow \{1, \dots, n\}$  is called a *labeling* of the vertices of  $G$ . The bandwidth of the labeling  $\phi$  is defined as

$$\max_{(i,j) \in E} |\phi(i) - \phi(j)|.$$

The bandwidth  $\sigma_\infty(G)$  of a graph  $G$  is the minimum of this number over all labelings, i.e.,

$$\sigma_\infty(G) := \min \left\{ \max_{(i,j) \in E} |\phi(i) - \phi(j)|; \phi : V \rightarrow \{1, \dots, n\} \right\}.$$

In terms of matrices, the bandwidth problem asks for a simultaneous permutation of the rows and columns of the adjacency matrix of  $G$  such that all nonzero entries are as close as possible to the main diagonal. The bandwidth problem is NP-hard [59].

The bandwidth problem originated in the 1950s from sparse matrix computations, and received much attention since Harary's description of the problem [37], and Harper's paper on the bandwidth of the  $n$ -cube [38]. The bandwidth problem arises in many different engineering applications which try to achieve efficient storage and processing. For more information on the bandwidth problem see, e.g. [13, 18, 42], and the references therein.

The bandwidth problem is related to the QAP. Indeed, let  $A$  be the adjacency matrix of  $G$  and  $B = (b_{ij})$  defined by

$$b_{ij} := \begin{cases} 1 & \text{for } |i - j| > k \\ 0 & \text{otherwise.} \end{cases}$$

Then, if an optimal value of the corresponding QAP (see (27.1)) is zero, then the bandwidth of  $G$  is at most  $k$ . Since it is difficult to solve QAPs in practice for sizes greater than 30, other approaches to derive bounds for the bandwidth of a graph are necessary.

Several lower bounds for the bandwidth of a graph are established in [40, 41, 61]. In those papers, the basic idea to obtain lower bounds on  $\sigma_\infty(G)$  is connecting the bandwidth minimization problem with the following graph partition problem. Let  $(S_1, S_2, S_3)$  be a partition of  $V$  with  $|S_i| = m_i$  for  $i = 1, 2, 3$ . The *min-cut* problem is:

$$\begin{aligned} \text{OPT}_{\text{MC}} &:= \min \sum_{i \in S_1, j \in S_2} a_{ij} \\ (\text{MC}) \quad &\text{s.t. } (S_1, S_2, S_3) \text{ partitions } V \\ &|S_i| = m_i, i = 1, 2, 3, \end{aligned}$$

where  $A = (a_{ij})$  is the adjacency matrix of  $G$ . Now, if  $\text{OPT}_{\text{MC}} > 0$  then

$$\sigma_\infty(G) \geq m_3 + 1, \tag{27.22}$$

as noted in [40, 41, 61].

Helmburg et al. [40] use the following relaxation of MC to compute a bound for  $\text{OPT}_{\text{MC}}$ :

$$\begin{aligned}
\text{OPT}_{\text{HW}} &:= \min \frac{1}{2} \text{tr}(\hat{A}XDX^T) \\
(\text{MC}_{\text{HW}}) \quad \text{s.t.} \quad & X^TX = \text{Diag}(m) \\
& Xu_3 = u_n \\
& X^Tu_n = m,
\end{aligned}$$

where  $m = [m_1, m_2, m_3]^T$ ,  $\hat{A} = A + \frac{1}{n}(u_n^T A u_n)I_n - \text{Diag}(A u_n)$ , and

$$D = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

They also prove that

$$\text{OPT}_{\text{HW}} = -\frac{1}{2}\mu_2\lambda_2 - \frac{1}{2}\mu_1\lambda_n, \quad (27.23)$$

where  $\lambda_2$  and  $\lambda_n$  denote the second smallest and the largest Laplacian eigenvalue of  $G$ , respectively, and values of  $\mu_1, \mu_2$  ( $\mu_1 \geq \mu_2$ ) are given by

$$\mu_{1,2} = \frac{1}{n} \left( -m_1 m_2 \pm \sqrt{m_1 m_2 (n - m_1)(n - m_2)} \right).$$

To show (27.23), the authors combine a projection technique for partitioning nodes of a graph [63] and a generalization of the Hoffman–Wielandt inequality (see e.g., [58]). By using (27.22) and (27.23), Helmberg et al. [40] derive the following result.

**Theorem 27.9 ([40]).** *Let  $G$  be an undirected graph on  $n$  vertices with at least one edge. Let  $\lambda_2$  and  $\lambda_n$  denote the second smallest and the largest Laplacian eigenvalue of  $G$ , respectively. Let  $\alpha = \lfloor n\lambda_2/\lambda_n \rfloor$ .*

1. If  $\alpha \geq n - 2$ , then  $G = K_n$  and  $\sigma_\infty(G) = n - 1$ .
2. If  $\alpha \leq n - 2$  and  $n - \alpha$  is even, then  $\sigma_\infty(G) \geq \alpha + 1$ .
3. Otherwise  $\sigma_\infty(G) \geq \alpha$ .

A proof of Theorem 27.9 that is based on interlacing of Laplacian eigenvalues, is given by Haemers [36].

Povh and Rendl [61] show that  $\text{MC}_{\text{HW}}$  is equivalent to the following SDP problem:

$$\begin{aligned}
\min \frac{1}{2} \text{tr}(D \otimes \hat{A})Y \\
(\text{MC}_{\text{PR}}) \quad \text{s.t.} \quad & \frac{1}{2} \text{tr}((E_{ij} + E_{ji}) \otimes I_n)Y = m_i \delta_{ij}, \quad 1 \leq i \leq j \leq 3 \\
& \text{tr}(J_3 \otimes E_{ii})Y = 1, \quad 1 \leq i \leq n \\
& \text{tr}(V_i^T \otimes W_j)Y = m_i, \quad 1 \leq i \leq 3, 1 \leq j \leq n \\
& \frac{1}{2} \text{tr}((E_{ij} + E_{ji}) \otimes J_n)Y = m_i m_j, \quad 1 \leq i \leq j \leq 3 \\
& Y \in \mathcal{S}_{3n}^+,
\end{aligned}$$

where  $V_i = e_i u_3^T \in \mathbb{R}^{3 \times 3}$  and  $W_j = e_j u_n^T \in \mathbb{R}^{n \times n}$ ,  $1 \leq i \leq 3$ ,  $1 \leq j \leq n$ . Note that adding the constraint  $X \geq 0$  to the relaxation MC<sub>HW</sub> leads to an intractable model, while MC<sub>PR</sub> does permit tractable refinements. Therefore, in [61] it is suggested to tighten MC<sub>PR</sub> by adding nonnegativity constraints. This leads to a strengthened SDP relaxation of the min-cut problem and the following result.

**Proposition 27.2.** *Let  $G$  be an undirected and unweighted graph. If for some  $m = (m_1, m_2, m_3)$  it holds that  $\text{OPT}_{\text{MC}} \geq \alpha > 0$ , then*

$$\sigma_\infty(G) \geq \max \left\{ m_3 + 1, m_3 + \lceil \sqrt{2\alpha} \rceil - 1 \right\}.$$

The numerical results in [61] show that this new bound is significantly better than the bounds obtained from the spectral bound.

**Acknowledgements** The author would like to thank Edwin van Dam for valuable discussions and careful reading of this manuscript. The author would also like to thank an anonymous referee for suggestions that led to an improvement of this chapter.

## References

1. Alizadeh, F.: Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optimiz.* **5**, 13–51 (1995)
2. Anstreicher, K.M.: Recent advances in the solution of quadratic assignment problems. *Math. Program. Ser. B* **97**, 27–42 (2003)
3. Anstreicher, K.M., Brixius, N.W.: A new bound for the quadratic assignment problem based on convex quadratic programming. *Math. Program. Ser. A* **89**, 341–357 (2001)
4. Anstreicher, K.M., Brixius, N.W., Linderoth, J., Goux, J.-P.: Solving large quadratic assignment problems on computational grids. *Math. Program. Ser. B* **91**, 563–588 (2002)
5. Anstreicher, K.M., Wolkowicz, H.: On Lagrangian relaxation of quadratic matrix constraints. *SIAM J. Matrix Anal. App.* **22**(1), 41–55 (2000)
6. Biswas, R., Hendrickson, B., Karypis, G.: Graph partitioning and parallel computing. *Parallel Comput.* **26**(12), 1515–1517 (2000)
7. Brixius, N.W., Anstreicher, K.M.: Solving quadratic assignment problems using convex quadratic programming relaxations. *Optim. Method Softw.* **16**, 49–68 (2001)
8. Brixius, N.W., Anstreicher, K.M.: The Steinberg wiring problem. In: Grötschel, M. (ed.) *The Sharpest Cut: The Impact of Manfred Padberg and His Work*. SIAM, Philadelphia, USA (2004)
9. Burer, S., Vandenbussche, D.: Solving lift-and-project relaxations of binary integer programs. *SIAM J. Optimiz.* **16**, 726–750 (2006)
10. Burkard, R., Karisch, S.E., Rendl, F.: QAPLIB – A quadratic assignment problem library. *J. Global Optim.* **10**, 391–403 (1997)
11. Burkard, R., Cela, E., Pardalos, P.M., Pitsoulis, L.: The quadratic assignment problem. In: Du, D.-Z., Pardalos, P. M., (ed.) *Handbook of Combinatorial Optimization*, Kluwer, Dordrecht, The Netherlands **3**, 241–337 (1999)
12. Burkard, R., Dell’Amico, M., Martello, S.: *Assignment Problems*. SIAM, Philadelphia, USA (2009)
13. Chinn, P.Z., Chvátalová, J., Dewdney, A.K., Gibbs, N.E.: The bandwidth problem for graphs and matrices – a survey. *J. Graph Theory* **6**, 223–254 (1982)

14. Cvetković, D., Čangalović, M., Kovačević-Vujčić, V.: Semidefinite programming methods for the symmetric traveling salesman problem. In: Proceedings of the 7th International IPCO Conference, Springer-Verlag, London, UK, 126–136 (1999)
15. Çela, E.: The Quadratic Assignment Problem: Theory and Algorithms. Kluwer, Dordrecht, The Netherlands (1998)
16. Dai, W., Kuh, E.: Simultaneous floor planning and global routing for hierarchical building-block layout. *IEEE T. Comput. Aided D.* **5**, 828–837 (1987)
17. Dantzig, G.B., Fulkerson, D.R., Johnson, S.M.: Solution of a large-scale traveling salesman problem. *Oper. Res.* **2**, 393–410 (1954)
18. Díaz, J., Petit, J., Serna, M.: A survey on graph layout problems. *ACM Comput. Surveys* **34**, 313–356 (2002)
19. De Klerk, E., Pasechnik, D.V., Sotirov, R.: On semidefinite programming relaxations of the traveling salesman problem. *SIAM J. Optim.* **19**(4), 1559–1573 (2008)
20. De Klerk, E., Sotirov, R.: Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. *Math. Program. Ser. A* **122**(2), 225–246 (2010)
21. De Klerk, E., Sotirov, R.: Improved semidefinite programming bounds for quadratic assignment problems with suitable symmetry. *Math. Program. Ser. A* (to appear)
22. De Klerk, E., Dobre, C.: A comparison of lower bounds for symmetric circulant traveling salesman problem. *Discrete. Appl. Math.* (to appear)
23. De Klerk, E., Pasechnik, D.V., Sotirov, R., Dobre, C.: On semidefinite programming relaxations of maximum k-section. *Math. Program. Ser. B* (to appear)
24. Dobre C.: Semidefinite programming approaches for structured combinatorial optimization problems. PhD thesis, Tilburg University, The Netherlands, (2011)
25. Donath, W.E., Hoffman, A.J.: Lower bounds for the partitioning of graphs. *IBM Journal of Res. Dev.* **17**, 420–425 (1973)
26. Ding, Y., Ge, D., Wolkowicz, H.: On Equivalence of Semidefinite Relaxations for Quadratic Matrix Programming. *Math. Oper. Res.*, **36**(1), 88–104 (2011)
27. Elshafei, A.N.: Hospital layout as a quadratic assignment problem. *Oper. Res. Quart.* **28**, 167–179 (1977)
28. Eschermann, B., Wunderlich, H.J.: Optimized synthesis of self-testable finite state machines. In: 20th International Symposium on Fault-Tolerant Computing (FFTCS 20), Newcastle upon Tyne, UK (1990)
29. Fiduccia, C.M., Mattheyses, R.M.: A linear-time heuristic for improving network partitions. *Proceedings of the 19th Design Automation Conference*, 175–181 (1982)
30. Frank, M., Wolfe, P.: An algorithm for quadratic programming. *Nav. Res. Log. Quart.* **3**, 95–110 (1956)
31. Garey, M.R., Johnson, D.S., Stockmeyer, L.: Some simplified NP-complete graph problems. *Theoret. Comput. Sci.* **1**(3), 237–267 (1976)
32. Ghaddar, B., Anjos, M.F., Liers, F.: A branch-and-cut algorithm based on semidefinite programming for the minimum  $k$ -partition problem. *Ann. Oper. Res.* (2008). doi:10.1007/s10479-008-0481-4
33. Gijswijt, D.: Matrix algebras and semidefinite programming techniques for codes. PhD thesis, University of Amsterdam, The Netherlands (2005)
34. Gilmore, P.C.: Optimal and suboptimal algorithms for the quadratic assignment problem. *SIAM J. Appl. Math.* **10**, 305–31 (1962)
35. Goemans, M.X., Rendl, F.: Combinatorial optimization. In: Wolkowicz, H., Saigal, R., Vandenberghe, L. (ed.) *Handbook of Semidefinite Programming: Theory, Algorithms and Applications*, Kluwer (2000)
36. Haemers, W.H.: Interlacing eigenvalues and graphs. *Linear Algebra Appl.* **227/228**, 593–616 (1995)
37. Harary, F.: Problem 16. In: Fiedler, M. (ed.) *Theory of graphs and its applications*, Czechoslovak Academy of Science, Prague (1967)

38. Harper, L.H.: Optimal assignments of numbers to vertices. *J. SIAM* **12**, 131–135 (1964)
39. Held, M., Karp, R.M.: The traveling-salesman problem and minimum spanning trees. *Oper. Res.* **18**(6), 1138–1162 (1970)
40. Helmburg, C., Rendl, F., Mohar, B., Poljak, S.: A spectral approach to bandwidth and separator problems in graphs. *Linear and Multilinear Algebra* **39**, 73–90 (1995)
41. Juvan, M., Mohar, B.: Laplace eigenvalues and bandwidth-type invariants of graphs. *J. Graph Theory* **17**(3), 393–407 (1993)
42. Lai, Y.L., Williams, K.: A survey of solved problems and applications on bandwidth, edgesum and profiles of graphs. *J. Graph Theory* **31**, 75–94 (1999)
43. Lengauer, T.: *Combinatorial Algorithms for Integrated Circuit Layout*. Wiley, Chichester (1990)
44. Karisch, S.E.: Nonlinear approaches for quadratic assignment and graph partition problems. PhD thesis, Technical University Graz (1995)
45. Karisch, S.E., Rendl, F.: Semidefinite programming and graph equipartition. In: *Topics in Semidefinite and Interior-Point Methods*, The Fields Institute for research in Mathematical Sciences, Communications Series, vol. 18, pp. 77–95. Providence, Rhode Island (1998)
46. Kernighan, B.W., Lin, S.: An efficient heuristic procedure for partitioning graphs. *Bell System Tech. J.* **49**, 291–307 (1970)
47. Krarup, J., Pruzan, P.M.: Computer-aided layout design. *Math. Program. Stud.* **9**, 75–94 (1978)
48. Krarup, J.: Quadratic Assignment. Saertryk af data, 3–27 (1972)
49. Laporte, G., Mercure, H.: Balancing hydraulic turbine runners: a quadratic assignment problem. *Eur. J. Oper. Res.* **35**, 378–382 (1988)
50. Lawler, E.: The quadratic assignment problem. *Manage. Sci.* **9**, 586–599 (1963)
51. Lengauer, T.: *Combinatorial Algorithms for Integrated Circuit layout*. Wiley, Chichester (1990)
52. Lisser, A., Rendl, F.: Graph partitioning using linear and semidefinite programming. *Math. Program. Ser. B* **95**(1), 91–101 (2003)
53. Lovász, L., Schrijver, A.: Cones of matrices and setfunctions and 01 optimization. *SIAM J. Optimiz.* **1**(2), 166–190 (1991)
54. Mobasher, A., Taherzadeh, M., Sotirov, R., Khandani, A.K.: A near maximum likelihood decoding algorithm for MIMO systems based on semidefinite programming. *IEEE Trans. on Info. Theory* **53**(11), 3869–3886 (2007)
55. Mobasher, A., Sotirov, R., Khandani, A.K.: Matrix-lifting SDP for detection in multiple antenna systems. *IEEE Trans. on Signal Proc.* **58**(10), 5178–5185 (2010)
56. Nugent, C.E., Vollman, T.E., Rumel, J.: An experimental comparison of techniques for the assignment of facilities to locations. *Oper. Res.* **16**, 150–173 (1968)
57. Nyström, M.: Solving certain large instances of the quadratic assignment problem: Steinberg's examples. Technical report, California Institute of Technology, CA (1999)
58. Overton, M.L., Womersley, R.S.: On the sum of the largest eigenvalues of a symmetric matrix. *SIAM J. Matrix Anal. Appl.* **13**, 41–45 (1992)
59. Papadimitriou, Ch.H.: The NP-Completeness of the bandwidth minimization problem. *Computing* **16**(3), 263–270 (1976)
60. Povh, J., Rendl, F.: Copositive and semidefinite relaxations of the quadratic assignment problem. *Discrete Optim.* **6**(3), 231–241 (2009)
61. Povh, J., Rendl, F.: A copositive programming approach to graph partitioning. *SIAM J. Optimiz.* **18**, 223–241 (2007)
62. Rendl, F., Sotirov, R.: Bounds for the quadratic assignment problem using the bundle method. *Math. Program. Ser. B* **109**(2/3), 505–524 (2007)
63. Rendl, F., Wolkowicz, H.: A projection technique for partitioning nodes of a graph. *Ann. Oper. Res.* **58**, 155–179 (1995)
64. Schrijver, A.: A comparison of the Delsarte and Lovász bounds. *IEEE Trans. Inform. Theory* **25**, 425–429 (1979)

65. Steinberg, L.: The backboard wiring problem: a placement algorithm. *SIAM Rev.* **3**, 37–50 (1961)
66. Zhao, Q., Karisch, S.E., Rendl, F., Wolkowicz, H.: Semidefinite programming relaxations for the quadratic assignment problem. *J. Comb. Optim.* **2**, 71–109 (1998)
67. Wolkowicz, H., Zhao, Q.: Semidefinite programming relaxations for the graph partitioning. *Discrete Appl. Math.* **96/97**, 461–479 (1999)

# Chapter 28

## Computational Approaches to Max-Cut

Laura Palagi, Veronica Piccialli, Franz Rendl, Giovanni Rinaldi,  
and Angelika Wiegele

### 28.1 Introduction

Max-Cut is one of the most studied combinatorial optimization problems because of its wide range of applications and because of its connections with other fields of discrete mathematics (see, e.g., the book by Deza and Laurent [10]). Like other interesting combinatorial optimization problems, Max-Cut is very simple to state.

An undirected graph  $G = (V, E)$  is given with a weight edge function  $w : E \rightarrow \mathbb{R}$  encoded in its weighted adjacency matrix  $A$ . The problem is to find a cut, i.e., an edge subset

$$\delta(S) = \{e = uv \in E : u \in S, v \notin S\},$$

---

L. Palagi (✉)

Dipartimento di Ingegneria informatica automatica e gestionale A. Ruberti, Università degli Studi di Roma “La Sapienza” - Via Ariosto, 25 - 0185 Rome, Italy  
e-mail: [laura.palagi@uniroma1.it](mailto:laura.palagi@uniroma1.it)

V. Piccialli

Dipartimento di Informatica, Sistemi e Produzione, Università degli Studi di Roma Tor Vergata, via del Politecnico 1 - 00133 Rome, Italy  
e-mail: [piccialli@disp.uniroma2.it](mailto:piccialli@disp.uniroma2.it)

F. Rendl

Alpen-Adria-Universität Klagenfurt - Universitätsstr. 65–67 - 9020 Klagenfurt, Austria  
e-mail: [franz.rendl@uni-klu.ac.at](mailto:franz.rendl@uni-klu.ac.at)

G. Rinaldi

Istituto di Analisi dei Sistemi ed Informatica “A. Ruberti”, CNR - viale Manzoni 30 - 00185, Rome, Italy  
e-mail: [rinaldi@iasi.cnr.it](mailto:rinaldi@iasi.cnr.it)

A. Wiegele

Alpen-Adria-Universität Klagenfurt - Universitätsstr. 65–67 - 9020 Klagenfurt, Austria  
e-mail: [angelika.wiegele@uni-klu.ac.at](mailto:angelika.wiegele@uni-klu.ac.at)

where  $S$  is a (possibly empty) subset of  $V$ , such that its total weight  $w(\delta(S)) = \sum_{uv \in \delta(S)} a_{uv}$  is maximized. Put it differently, the problem asks for a bipartition  $(S, V \setminus S)$  of the vertices  $V$  of  $G$ , so that the total weight of the edges that go across the sets of the bipartition is maximal.

The problem can be restated as an unconstrained problem on binary variables with arbitrary quadratic objective function. Despite its simple formulation, this is an NP-hard problem, therefore some pseudo-enumeration technique is essentially unavoidable if one wants to solve it to optimality. To limit the growth of the number of subproblems that such a technique typically generates, good upper and lower bounds on the optimal solution value are necessary. Upper bounds require the definition and the solution of a suitable relaxation to the problem, while lower bounds are computable by finding a “good” cut with a heuristic algorithm.

The first computational experiments on non-trivial Max-Cut instances were carried out with Linear Programming (LP) based relaxations. More recently, as soon as the field of Semidefinite Programming (SDP) started to develop, SDP relaxations to the problem were used. The successful results obtained with SDP techniques in the efficient generation of both upper and lower bounds rapidly made Max-Cut the application of choice of SDP algorithms and greatly increased the interest on this problem among the members of the SDP community. This is the main motivation for the topic of this chapter in which some recent methods are surveyed with some emphasis on their impact on the computation of heuristic and exact solutions. To be consistent with the topic of this volume, the chapter mostly covers aspects of the SDP relaxation of Max-Cut, while the basic LP based relaxations are only briefly outlined.

The following simple semidefinite program plays a fundamental role in combinatorial optimization. For a given matrix  $C$  in the space of the  $n \times n$  symmetric matrices  $\mathcal{S}^n$ , find  $X \in \mathcal{S}^n$  such that

$$z_P = \max\{\langle C, X \rangle : \text{diag}(X) = e, X \succeq 0\}, \quad (28.1)$$

where  $e$  is the vector of all ones,  $\text{diag}(X)$  is the vector containing the diagonal of  $X$  and  $X \succeq 0$  indicates that  $X$  is positive semidefinite.

The associated dual problem is given as follows:

$$z_D = \min\{e^T y : \text{Diag}(y) - C \succeq 0\}, \quad (28.2)$$

where  $\text{Diag}(y)$  is the diagonal matrix having as a diagonal the vector  $y \in \mathbb{R}^n$ . Since both the primal and the dual problem have strictly feasible points ( $X = I$ , the identity matrix, for the primal and  $y = (\lambda_{\max}(C) + 1)e$  for the dual, where  $\lambda_{\max}(C)$  denotes the largest eigenvalue of  $C$ ), both the primal and the dual optimal values are attained and are equal to each other. Problem (28.1) is perhaps best known in connection with Max-Cut. To formulate the problem, we define the Laplacian  $L = \text{Diag}(Ae) - A$  that is associated with the weighted adjacency matrix  $A$  of  $G$ . Max-Cut can then be formulated as follows:

$$z_{MC} = \max \left\{ \frac{1}{4} x^T L x : x \in \{-1, 1\}^n \right\}. \quad (28.3)$$

Bipartitions  $(S, V \setminus S)$  are encoded by  $n$ -dimensional vectors  $x$  with  $x_i = 1$  for  $i \in S$  and  $x_i = -1$  for  $i \notin S$ . It is a well-known fact and easy to verify that in this case

$$\sum_{uv \in \delta(S)} a_{uv} = \frac{1}{4} x^T L x.$$

Since  $x^T L x = \langle L, xx^T \rangle$  and  $xx^T \succeq 0$  with  $\text{diag}(xx^T) = e$ , it is clear that problem (28.1) with  $C = \frac{1}{4}L$  provides a relaxation of Max-Cut.

The results presented on Max-Cut, and those on the related problems described in Sect. 28.2, all have some connection to the simple semidefinite program (28.1).

In Sect. 28.3 the main methods are described that are used to solve (28.1). In particular, Sect. 28.3.1 is devoted to the classic solution method, the one based on the interior-point approach. Aiming at finding methods that are better suited for problems with a sparse objective function matrix, some reformulations of (28.1) and of its dual (28.2) have been recently studied that avoid semidefinite matrices and turn out to be quite useful from a computational point of view.

A first natural way to eliminate the semidefiniteness constraint  $\text{Diag}(y) - C \succeq 0$  in the dual problem is as follows. We first observe that the primal constraints  $\text{diag}(X) = e$  imply  $\text{tr}(X) = n$ , where  $\text{tr}(X)$  is the trace of matrix  $X$ . Adding this redundant constraints to (28.1) leads to the following dual problem, with additional dual variable  $\lambda$  for the redundant constraint:

$$\min\{e^T y + n\lambda : \lambda I + \text{Diag}(y) - C \succeq 0\}.$$

For any feasible solution of this problem we have

$$\lambda_{\max}(C - \text{Diag}(y) - \lambda I) \leq 0,$$

hence

$$\lambda \geq \lambda_{\max}(C - \text{Diag}(y)).$$

Therefore, minimizing  $e^T y + n\lambda$  forces equality and yields the following equivalent formulation of the dual:

$$\min_{y \in \mathbb{R}^n} e^T y + n\lambda_{\max}(C - \text{Diag}(y)).$$

We can slightly simplify this problem by removing the linear term. We represent  $y \in \mathbb{R}^n$  through its components parallel and orthogonal to  $e$ ,

$$y = \alpha e + v \text{ where } \langle v, e \rangle = 0 \text{ and } \alpha \in \mathbb{R}. \quad (28.4)$$

In this case the dual simplifies to

$$z_D = \min\{n\lambda_{\max}(C - \text{Diag}(v)) : \langle v, e \rangle = 0\}. \quad (28.5)$$

This is a convex but non-smooth minimization problem, which can be solved with subgradient techniques from convex optimization. In Sect. 28.3.2 we briefly discuss the spectral bundle method, which is tailored to solve problems of this type.

A second reformulation is based on the observation that  $X \succeq 0$  is equivalent to  $X = VV^T$  for some  $n \times r$  matrix  $V$ . We denote the columns of  $V^T$  by  $v_i$  for  $i = 1, \dots, n$ , i.e.,  $V^T = (v_1, \dots, v_n)$  and  $v_i \in \mathbb{R}^r$ . In this case,  $\text{diag}(VV^T) = e$  is equivalent to  $\|v_i\|^2 = 1$  for  $i = 1, \dots, n$ . This gives the non-linear problem

$$\max \left\{ \sum_{ij} c_{ij} \langle v_i, v_j \rangle : v_i \in \mathbb{R}^r, \|v_i\|^2 = 1, i = 1, \dots, n \right\}. \quad (28.6)$$

This formulation (in minimization form) is investigated in Sect. 28.3.3, and is the basis for the famous Goemans-Williamson hyperplane rounding technique that will be described in Sect. 28.4 (see also Chap. 6 of this Handbook). Because of their computational efficiency most of Sect. 28.3 is devoted to the methods based on this reformulation.

The methods described in Sect. 28.3 are used in Sect. 28.4 to find approximate solutions to Max-Cut and in Sect. 28.5 to solve the problem to optimality. Finally, some computational results for the algorithms described in these two sections are reported in Sect. 28.6. The results of the last few years obtained by exploiting the reformulation (28.6) are perhaps the most interesting achievements, as far as the computation of an optimal solution to (28.3) is concerned. For this reason, a large part of this section is devoted to them.

## 28.2 Extensions

In this section we will describe several types of problems which all have semidefinite relaxations containing the basic problem (28.1) in addition to other constraints.

### 28.2.1 Max- $k$ -Cut and Coloring

Max- $k$ -Cut and Coloring can be viewed as partition problems in graphs. Max- $k$ -Cut takes the weighted adjacency matrix  $A$  of a weighted graph  $G$  as input and asks to partition the vertices  $V$  of  $G$  into  $k$  sets  $(S_1, \dots, S_k)$  so as to maximize the weight of all edges joining distinct partition blocks. Coloring, or more specifically  $k$ -Coloring, asks to partition  $V$  into (at most)  $k$  stable sets of  $G$ .

It seems natural to represent the partition  $(S_1, \dots, S_k)$  of  $V$  by the incidence vectors  $x_i$  of  $S_i$  which are collected in the partition matrix  $X = (x_1, \dots, x_k)$ . Since each vertex in  $V$  is in exactly one of the sets, the row sums of this matrix give the all-ones vector  $e$ . Hence  $k$ -partitions of  $V$  are in one-to-one correspondence with the set

$$\mathcal{P}_k = \{X : X = (x_{ij})_{n \times k}, x_{ij} \in \{0, 1\}, Xe = e\}.$$

If the cardinalities of the partition blocks are also specified, we get additional (linear) constraints on the column sums of  $X$ .

Furthermore, for a  $k$ -partition matrix  $X$  we have  $\text{diag}(XX^T) = e$  and  $kXX^T - J \geq 0$ , where  $J = ee^T$ .

The block-diagonal structure of  $XX^T$  also shows that  $\langle A, XX^T \rangle$  is twice the weight of edges inside partition blocks. Using the Laplacian  $L$ , associated to  $A$ , it follows that

$$\frac{1}{2}\langle L, XX^T \rangle$$

is the weight of edges joining distinct partition blocks. This leads to the following semidefinite relaxation for Max- $k$ -Cut after the variable transformation  $Y = XX^T$ :

$$\max \left\{ \frac{1}{2}\langle L, Y \rangle : \text{diag}(Y) = e, Y - \frac{1}{k}J \geq 0, Y \geq 0 \right\}.$$

Finally, the transformation  $Z = \frac{k}{k-1}(Y - \frac{1}{k}J)$  leads to

$$\max \left\{ \frac{k}{2(k-1)}\langle L, Z \rangle : \text{diag}(Z) = e, Z \geq 0, z_{ij} \geq -\frac{1}{k-1} \text{ for } i < j \right\}.$$

This formulation contains again the constraints  $\text{diag}(Z) = e, Z \geq 0$  from the basic relaxation (28.1). Frieze and Jerrum [13] adopted the Goemans–Williamson hyperplane rounding idea for this relaxation and provided performance guarantees  $\alpha_k$ , which were slightly improved later by De Klerk et al. [9]. Here are these ratios for small  $k$ :  $\alpha_5 \approx 0.87661$ ,  $\alpha_4 \approx 0.85749$ ,  $\alpha_3 \approx 0.83601$  and  $\alpha_2 = 0.87856$ , which corresponds to the Goemans–Williamson ratio for Max-Cut. From a practical point of view, we refer to Ghaddar et al. [14] who provide computational experience for the Max- $k$ -Cut problem using relaxations based on semidefinite optimization in combination with branch and cut.

Turning to  $k$ -Coloring, we again look for a  $k$ -partition  $(S_1, \dots, S_k)$ , but in this case each  $S_i$  also has to be a stable set. Thus, if  $x_i$  represents the stable set  $S_i$ , we need to impose the constraint

$$(x_i)_u(x_i)_v = 0 \text{ for all } uv \in E(G),$$

to insure that for the edges  $uv$ , we cannot have both endpoints in  $S_i$ .

The smallest  $k$  such that the graph  $G$  has a  $k$ -Coloring is usually called the *chromatic number*  $\chi(G)$  of  $G$ . We therefore have

$$\chi(G) \leq \min \left\{ k : Y - \frac{1}{k}J \geq 0, \text{diag}(Y) = e, y_{uv} = 0 \text{ for all } uv \in E(G) \right\}.$$

Using the above transformation again, we get

$$\min\{\alpha : \text{diag}(Z) = e, Z \succeq 0, z_{uv} = \alpha \text{ for all } uv \in E(G)\}.$$

We point out once more that this relaxation has the basic structure of (28.1). Karger et al. [30] use this model to adopt the hyperplane rounding idea to get approximation results for Coloring. They also show that this model corresponds, in disguised form, to the Lovász Theta number, see [34].

### 28.2.2 Ordering Problem

Ordering problems assign a profit to each ordering (of  $n$  objects) and ask for an ordering of maximum profit. In the simplest case, the profit of an ordering is simply built up by a profit vector  $c = (c_{ij}), i \neq j$ . In this case the profit is given by  $\sum_{i < j} c_{ij}$  where the summation is over all terms  $i$  before  $j$  in the ordering. This type of problem is called *linear ordering problem*. (See also Chap. 29 of this Handbook.)

Since we will consider matrix relaxations of this problem, it is natural to consider cost functions that depend quadratically on the ordering. In other words, we allow the profit of an ordering to be built up by terms  $c_{ij,rs}$ , provided  $i$  is before  $j$ , and  $r$  is before  $s$  in the ordering. We call this a *quadratic ordering problem*. Since we may encode orderings by permutations, this problem asks for a permutation of the elements of the set  $S = \{1, \dots, n\}$  such that some quadratic objective function is maximized.

In order to formulate the problem we introduce the following notation. Let  $\mathcal{P}_n$  be the set of all permutations of the elements of  $S$ . For a permutation  $\pi \in \mathcal{P}_n$ , we define a characteristic vector  $\chi(\pi) \in \{0, 1\}^{\binom{n}{2}}$  of the underlying ordering as follows: for all  $1 \leq i < j \leq n$ , we set

$$\chi(\pi)_{ij} = \begin{cases} 1 & \text{if } \pi(i) < \pi(j) \\ 0 & \text{otherwise.} \end{cases}$$

Let  $\text{LO}(n)$  denote the (linear) ordering polytope on  $n$  elements, i.e., the polytope

$$\text{LO}(n) = \text{conv}\{\chi(\pi) \mid \pi \in \mathcal{P}_n\} \subseteq \mathbb{R}^{\binom{n}{2}}.$$

The quadratic ordering problem can then be stated as

$$\max \left\{ x^\top C x : x \in \text{LO}(n) \cap \{0, 1\}^{\binom{n}{2}} \right\}, \quad (28.7)$$

where  $C$  is a real  $\binom{n}{2} \times \binom{n}{2}$  matrix. For orderings clearly transitivity must hold, i.e., if  $i$  is before  $j$  and  $j$  is before  $k$  then  $i$  must be before  $k$ . To model transitivity,

it is sufficient to forbid directed 3-cycles. This is achieved by asking that the 3-dicycle inequalities

$$0 \leq x_{ij} + x_{jk} - x_{ik} \leq 1 \quad (28.8)$$

hold for all  $i < j < k$  [22, 38].

When moving to matrix relaxations, it turns out to be more convenient to model the characteristic vectors of orderings through variables taking the values -1 and 1. The variable transformation  $y_{ij} = 2x_{ij} - 1$  leads to the following form of the 3-dicycle inequalities

$$-1 \leq y_{ij} + y_{jk} - y_{ik} \leq 1.$$

Since the  $y_{ij}$ 's are -1 or 1, we have in fact that

$$|y_{ij} + y_{jk} - y_{ik}| = 1.$$

This thus shows that inequalities (28.8) are satisfied for  $x \in \{0, 1\}^{\binom{n}{2}}$  if and only if the equations

$$y_{ij}y_{jk} - y_{ij}y_{ik} - y_{ik}y_{jk} = -1 \quad (28.9)$$

hold for all  $i < j < k$ , and  $y_{ij} = 2x_{ij} - 1$ . In [7] it is shown that the quadratic ordering problem can be formulated as a Max-Cut problem of a graph on  $\binom{n}{2} + 1$  nodes with additional constraints (28.9). In particular, it is shown that these constraints induce a face of the cut polytope. Thus, a relaxation of the quadratic ordering problem in  $\{-1, 1\}$  variables  $y_{ij}$  is obtained by weakening the condition  $Y = yy^T$  to  $Y - yy^T \geq 0$ .

This last condition is equivalent to  $Z = \begin{pmatrix} 1 & y^T \\ y & Y \end{pmatrix} \geq 0$ . Hence we get

$$\max \left\{ \langle Q, Z \rangle : \text{diag}(Z) = e, -z_{ij,ik} - z_{ik,jk} + z_{ij,jk} = -1, Z \succeq 0 \right\},$$

with suitably chosen cost matrix  $Q$ . Note that matrix  $Z$  is of dimension  $\binom{n}{2} + 1$ , i.e., it is a matrix indexed by all pairs  $(i, j)$ ,  $i < j$ , plus an additional row and column. This again is the semidefinite program (28.1) with some additional linear constraints.

Clearly, algorithms for solving the Max-Cut problem can be used for solving the quadratic ordering problem. An extended version of the algorithm BiqMac (see Sect. 28.5) has been used to solve special instances of this problem arising from bipartite crossing minimization. This extended version is capable of treating several additional constraints, in our case equalities (28.9). In [7] it is demonstrated that using this semidefinite programming based algorithm clearly outperforms other existing approaches, like standard linearization or branch-and-cut. We also refer to [28] for recent strengthenings and extensions of this approach.

## 28.3 How to Solve the Semidefinite Programs

Semidefinite optimization problems have their nonlinearity only in the constraint that the primal matrix  $X$  should be contained in the closed, convex cone of semidefinite matrices. Newton method is the basis for a family of interior-point based algorithms to solve SDP. We briefly recall their key features in Sect. 28.3.1 below. The reformulation of the dual problem as eigenvalue optimization problem (28.5) is the starting point for bundle, and more specifically, spectral bundle methods. These will be briefly described in Sect. 28.3.2 below. Finally, the primal non-linear model (28.6) is the starting point for the non-linear programming based algorithms described in Sect. 28.3.3 below.

### 28.3.1 Interior-Point Methods

The key idea to solve semidefinite optimization problems by interior-point methods consists in applying the Newton method to the optimality conditions of the primal-dual pair of problems. We consider the following standard form of the primal-dual pair:

$$\min\{\langle C, X \rangle : \mathcal{A}(X) = b, X \succeq 0\} = \max\{b^T y : Z = C - \mathcal{A}^T(y) \succeq 0\},$$

where the linear operator  $\mathcal{A}(\cdot)$  maps (symmetric) matrices to  $\mathbb{R}^m$ , with  $(\mathcal{A}(X))_i = \langle \mathcal{A}_i, X \rangle$ , and its adjoint is  $\mathcal{A}^T y = \sum_i \mathcal{A}_i y_i$ .

The dimension of the matrix space is  $n$ , and  $b \in \mathbb{R}^m$ . We assume that both problems have strictly feasible points, ensuring that both the primal and the dual optimal values are attained and are equal to each other. Interior-point methods approximately solve the parametrized optimality conditions

$$\mathcal{A}(X) = b, \quad Z + \mathcal{A}^T(y) = C, \quad ZX = \mu I, \quad \mu > 0$$

for fixed  $\mu > 0$ , and iterate for decreasing values of  $\mu$ , until  $\mu \approx 0$ . The main computational effort in each iteration consists in setting up and solving a linear system for the update  $\Delta y$  of the dual variables. This involves several operations of order  $O(n^3)$ , such as computing  $Z^{-1}$  explicitly and performing several matrix multiplications of generally dense matrices of order  $n$  to determine the updates  $\Delta X$  and  $\Delta Z$ . Having these updates, a line-search is carried out to arrive at a new iterate of positive definite matrices.

The linear system to be solved in case of (28.1) is of order  $n$  and has the simple coefficient matrix  $X \circ Z^{-1}$ .

A representative table that provides a flavor of the computational effort of a dual-scaling based interior-point method particularly suitable for Max-Cut is reported in Sect. 28.6.

For a complete review of recent advances in interior-point methods we refer the reader to the Chap. 17 and the Chap. 23 of this Handbook.

### 28.3.2 Spectral Bundle Method

The formulation (28.5) of the basic semidefinite problem (28.1) captures the nonlinearity of the problem in the eigenvalue function. The Spectral Bundle method SB from [25] is tailored to solve problems of the following type. For given  $C, A_1, \dots, A_m \in \mathcal{S}_n$  and  $b \in \mathbb{R}^m$ , define

$$f(y) = b^T y + \lambda_{\max}(C - \sum_i y_i A_i)$$

and consider

$$\min_{y \in \mathbb{R}^m} f(y).$$

In [25] it is pointed out that this problem is equivalent to the semidefinite program

$$\min\{b^T y : A^T y - C \geq 0\}$$

in case that the identity matrix  $I$  is contained in the linear hull, spanned by the matrices  $A_i$ . It is clear that our problem (28.2) falls into this class of problems.

The spectral bundle method exploits the following easy facts. If  $M \in \mathcal{S}^n$  then

$$\lambda_{\max}(M) = \max_{p^T p=1} p^T M p = \max_{W \in \text{conv}\{pp^T : p^T p=1\}} \langle M, W \rangle = \max_{W \geq 0, \text{tr}(W)=1} \langle M, W \rangle.$$

The bundle method generates a sequence of iterates that approaches the minimizer of  $f(y)$ . Suppose the current iterate is  $\hat{y}$ . Let  $\hat{p}$  be a unit eigenvector to  $\lambda_{\max}(C - A^T(\hat{y}))$ . The main idea of the spectral bundle method consists in replacing the set  $\{W : \text{tr}(W) = 1, W \geq 0\}$  by a suitable subset

$$S_P = \{W = PVP^T : \text{tr}(V) = 1, V \geq 0\},$$

defined through the “bundle”  $P$ , which is an  $n \times k$  matrix satisfying  $P^T P = I_k$ . The number  $k$  of columns in  $P$  can be selected by the user, and is typically much smaller than  $n$ . It is clear that

$$\lambda_{\max}(M) = \max_{W \in S_I} \langle M, W \rangle \geq \max_{W \in S_P} \langle M, W \rangle.$$

We define

$$\hat{f}(y) = b^T y + \max_{W \in S_P} \langle C - A^T(y), W \rangle \leq f(y).$$

The idea now is to select the matrix  $P$  in such a way that  $\hat{f}(y)$  approximates  $f(y)$  reasonably well around  $\hat{y}$ . We therefore assume that  $\hat{p}$  is among the columns of  $P$ , ensuring that  $f(\hat{y}) = \hat{f}(\hat{y})$ . To make sure that the next iterate does not take us too

far from the current iterate, the spectral bundle method minimizes  $\hat{f}(y)$  incremented with a regularization term

$$\min_y \hat{f}(y) + \frac{1}{2t} \|y - \hat{y}\|^2.$$

The regularization parameter  $t > 0$  is a user defined constant. Upon inserting the definition of  $\hat{f}(y)$  we get a Min-Max problem

$$\min_y \max_{W \in S_P} b^T y + \langle C - A^T(y), W \rangle + \frac{1}{2t} \|y - \hat{y}\|^2.$$

In [25] it is shown that this can be solved by exchanging Min and Max, leading to the update

$$y = \hat{y} - t(b - A(PV P^T)),$$

with (yet) unknown  $V$ . To get  $V$ , one back-substitutes  $y$  to get the quadratic semidefinite program

$$\max_{W \in S_P} b^T \hat{y} + \langle C - A^T(\hat{y}), W \rangle - \frac{t}{2} \|b - A(W)\|^2,$$

where  $W = PV P^T$ . This quadratic semidefinite problem (with just one scalar equation  $\text{tr}(V) = 1$ ) can be solved by standard interior-point methods to yield  $V$ . This determines the new trial point

$$y_{\text{new}} = \hat{y} - t(b - A(PV P^T)).$$

The function  $f$  is now evaluated at the new point, the bundle  $P$  is updated (see [25] for details) and a new iteration is started. The main computational steps in one iteration are the solution of the quadratic semidefinite program to get  $V$  and the evaluation of the function at the new trial point, which amounts to an eigenvalue computation. A table with some representative results of this method applied to the dual problem (28.2) is reported in Sect. 28.6.

### 28.3.3 Non-linear Programming Methods

In this section we review the Non-Linear Programming (NLP) based approaches for solving problem (28.1). Most of these algorithms exploit the special structure of the constraints of the problem in order to define non-linear programming reformulations.

We refer to a minimization problem, so that we define  $Q = -C$  and we consider

$$\min_{X \in \mathbb{R}^{n \times n}} \{\langle Q, X \rangle : \text{diag}(X) = e, X \geq 0\}. \quad (28.10)$$

This SDP relaxation of problem (28.3) was first derived by Goemans-Williamson in [15], by replacing each variable  $x_i$  of problem (28.3) with a vector  $v_i \in \mathbb{R}^n$  (or  $\mathbb{R}^r$  with  $r \leq n$ ) obtaining problem (28.6). Since one can show the existence of a low rank optimal solution of (28.10) and any given matrix  $X \geq 0$  with rank  $r$  can be written as  $X = VV^T$  for some  $n \times r$  matrix  $V$ , the positive semidefiniteness constraint can be eliminated, and problem (28.10) reduces to the so-called Low Rank SDP formulation (LRSDP)

$$\min_{V \in \mathbb{R}^{n \times r}} \{\langle Q, VV^T \rangle : \langle E_{ii}, VV^T \rangle = 1, i = 1, \dots, n\}, \quad (28.11)$$

which turns out to be a non-convex non-linear problem ( $E_{ii}$  denotes the  $n \times n$  matrix with the  $i$ -th diagonal component equal to one and all the other components equal to 0).

A global minimizer of problem (28.11) is a solution of problem (28.10) provided that  $r$  is not less than the rank  $r_{\min}$  of a minimum rank optimal solution of problem (28.10). Although the value of  $r_{\min}$  is not known, an upper bound can be easily computed by exploiting the result proved in [4, 21, 35], that gives

$$r_{\min} \leq \widehat{r} = \frac{\sqrt{8n+1}-1}{2}. \quad (28.12)$$

In principle, the value  $\widehat{r}$  guarantees the correspondence between solutions of (28.10) and global solutions of (28.11).

However, even assuming that a suitable value of  $r$  is known, in order to solve problem (28.10), a global minimizer of the non-convex problem (28.11) is needed. In general this is a hard task, tied to the definition of suitable global optimality conditions. Thanks to strong duality and exploiting the connection between (28.1) and its dual (28.2), in [18, 19, 29] a global optimality condition has been proved. Indeed, let  $r \geq r_{\min}$ . A feasible matrix  $V^* \in \mathbb{R}^{n \times r}$  is a global minimizer of the LRSDP problem (28.11) if and only if

$$(Q + \text{Diag}(\lambda^*))V^* = 0 \text{ and } Q + \text{Diag}(\lambda^*) \geq 0,$$

where  $\lambda^* \in \mathbb{R}^n$  is uniquely defined component-wise as

$$\lambda_i^* = \lambda_i(V^*) = -\langle E_{ii}Q, V^*V^{*T} \rangle \quad i = 1, \dots, n. \quad (28.13)$$

The LRSDP reformulation combined with the above optimality condition leads to the Incremental Rank Algorithm (IRA) for the solution of the SDP problem (28.10) reported below, that encompasses most of the algorithms proposed in the literature.

### Incremental Rank Algorithm (IRA)

**Data:**  $Q$  and integer  $p \geq 1$ .

**Initialization.** Set integers  $2 \leq r^1 < r^2 < \dots < r^p$  with  $r^p \in [\widehat{r}, n]$  where  $\widehat{r}$  is given by (28.12). Choose  $\varepsilon > 0$ .

**For**  $j = 1, \dots, p$

**S.0** Set  $r = r^j$  in problem (28.11), choose  $V^0 = V^{0j} \in \mathbb{R}^{n \times r^j}$ .

**S.1** Starting from  $V^0$ , find a stationary point  $\widehat{V} \in \mathbb{R}^{n \times r^j}$  of problem (28.11).

**S.2** Compute the minimum eigenvalue  $\lambda_{\min}(Q + \text{Diag}(\lambda(\widehat{V})))$  of  $Q + \text{Diag}(\lambda(\widehat{V}))$ .

**S.3** If  $\lambda_{\min}(Q + \text{Diag}(\lambda(\widehat{V}))) \geq -\varepsilon$ , then **exit**.

**End**

**Return**  $\widehat{V} \in \mathbb{R}^{n \times r^j}$ ,  $\lambda_{\min}(Q + \text{Diag}(\lambda(\widehat{V})))$  and

$$z_{LB} = \langle Q, \widehat{V}\widehat{V}^T \rangle + n \min\{0, \lambda_{\min}(Q + \Lambda(\widehat{V}))\}$$

The IRA scheme returns  $\widehat{V}$ ,  $\lambda_{\min}(Q + \text{Diag}(\lambda(\widehat{V})))$  and  $z_{LB}$ . If  $\lambda_{\min}(Q + \text{Diag}(\lambda(\widehat{V}))) \geq -\varepsilon$ , then the matrix  $Q + \text{Diag}(\lambda(\widehat{V}))$  is positive semidefinite within a tolerance  $\varepsilon$  so that a solution of (28.10) is obtained as  $X^* = \widehat{V}\widehat{V}^T$ . It is worth noting that, even if the optimality condition is not met, since  $(\lambda(\widehat{V}) + \lambda_{\min}(Q + \text{Diag}(\lambda(\widehat{V})))e)$  is feasible for the dual of problem (28.10), the value  $z_{LB} = \langle Q, \widehat{V}\widehat{V}^T \rangle + n\lambda_{\min}(Q + \text{Diag}(\lambda(\widehat{V})))$  provides a lower bound on the solution of problem (28.10) (see e.g. [19, 36] and the introduction of this chapter).

In principle, the IRA scheme may fail to produce a solution of problem (28.10) because the algorithm at Step S.1 may converge to stationary points which are not global minimizers of problem (28.11) even when  $r = \widehat{r}$ . Although this problem does not arise in practice, the drawback can be theoretically overcome by using an optimality result from [29]. Indeed, in that paper it has been proved that a rank deficient  $\widehat{V}$  satisfying the second order necessary condition for problem (28.11) is necessarily a global minimizer. Hence, allowing an increase of  $r$  up to  $n$  and using an algorithm converging to second order points at Step S.1, the IRA scheme is guaranteed to converge to a global minimizer of problem (28.11).

What differentiates the various algorithms proposed in the literature is how to compute a stationary point of the LRSDP problem (28.11) at Step S.1, and whether the optimality check at Step S.2 is explicitly performed or not. In most papers, the computation of a stationary point is achieved by means of an unconstrained reformulation of problem (28.11).

In [8] the classical Augmented Lagrangian method [26, 37] is used for solving the LRSDP formulation of a general SDP problem with linear constraints. In particular, a sequence of minimizations of the Augmented Lagrangian function (specialized to problem (28.11))

$$\begin{aligned}\mathcal{L}(V, \lambda^k; \sigma^k) = & \langle Q, VV^T \rangle + \sum_{i=1}^n \lambda_i^k (\langle E_{ii}, VV^T \rangle - 1) \\ & + \frac{\sigma^k}{2} \sum_{i=1}^n (\langle E_{ii}, VV^T \rangle - 1)^2\end{aligned}\quad (28.14)$$

is performed for suitable values of  $(\lambda^k, \sigma^k)$ , where  $\sigma^k$  is increasing and  $\lambda^k$  is obtained with some updating rule.

In [19], Grippo et al. exploit the structure of the constraints in (28.11) to define an exact penalty function  $P(V)$ . The penalty function is obtained by replacing the multipliers  $\lambda$  in the augmented Lagrangian function  $\mathcal{L}(V, \lambda; \sigma)$  with the closed expression (28.13), and by fixing the value of the penalty parameter to a computable value  $\bar{\sigma} > 0$ , so that  $P(V) = \mathcal{L}(V, \lambda(V); \bar{\sigma})$ . Thus, a single unconstrained minimization of the twice continuously differentiable function  $P(V)$  provides a stationary point of problem (28.11).

More recently, Journée et al. [29] use a trust region method for the optimization over a manifold derived from [1] which relies on a particular quotient manifold. Their algorithm is defined for a slightly more general SDP problem than (28.10) but relies on exploiting the special structure of the constraints to find a closed expression of the multipliers which in the special case of problem (28.11) returns (28.13).

For the particular class of SDP problem (28.10) arising as a Max-Cut relaxation, Burer and Monteiro in the computational section of [8] use a different unconstrained formulation, based on a sort of Rayleigh quotient. The original idea goes back to Homer and Peinado [27], where the change of variables  $x_{ij} = \langle v_i, v_j \rangle / \|v_i\| \|v_j\|$  for the elements of  $X$  with  $v_i \in \mathbb{R}^n$ ,  $i = 1, \dots, n$  has been used to formulate an unconstrained optimization problem equivalent to the original (28.11). Their approach leads to a problem of dimension  $n^2$  which is solved by a parallel gradient method but turns out to be impractical for large values of  $n$ . Burer and Monteiro combine this approach with the low rank idea by replacing vectors  $v_i \in \mathbb{R}^n$  with vectors  $v_i \in \mathbb{R}^r$ , for increasing values of  $r$  up to  $\bar{r}$ , getting the unconstrained problem

$$\min f_r(V) = \sum_{i=1}^n \sum_{j=1}^n q_{ij} \frac{\langle v_i, v_j \rangle}{\|v_i\| \|v_j\|}, \quad v_i \in \mathbb{R}^r. \quad (28.15)$$

The practical performance of this approach is pretty good, but the underlying convergence theory is not deeply investigated in [8]. It is easy to show the equivalence between the unconstrained problem (28.15) and problem (28.11) (see [16]). However, standard convergence results are not immediately applicable, since continuous differentiability of the objective function over the whole space  $\mathbb{R}^{nr}$  is required, while the objective function of (28.15) is not even defined at points where  $\|v_i\| = 0$  for at least one index  $i$ , and compactness of the level sets is missing as well. To overcome these drawbacks, Grippo et al. propose in [16, 17] a modification

of the objective function of problem (28.15) which uses an additive shifted barrier penalty term. The resulting unconstrained problem is

$$\min_{V \in S_\delta} f_r(V) + \tau \sum_{i=1}^n \frac{(\|v_i\|^2 - 1)^2}{\delta^2 - \max\{1 - \|v_i\|^2, 0\}^2} \quad (28.16)$$

where  $S_\delta \equiv \{V \in \mathbb{R}^{n \times r} : \|v_i\|^2 > 1 - \delta, \quad i = 1, \dots, n\}$ , and  $0 < \delta < 1$  and  $\tau > 0$  are fixed scalars. Problem (28.16) has a continuously differentiable objective function on a suitable open set and compact level sets; moreover, equivalence to problem (28.11) still holds.

A performance comparison among all the approaches described here is presented in Sect. 28.6.

## 28.4 How to Find Good Max-Cut Solutions

As it was mentioned in the Sect. 28.1, an instance of Max-Cut defined by the weighted adjacency matrix  $A \in \mathbb{R}^{n \times n}$  of a graph  $G = (V, E)$  with  $n = |V|$  nodes is equivalent to an instance of an *unconstrained binary quadratic program*

$$\max\{x^T Q x : x \in \{0, 1\}^{n-1}\}, \quad (28.17)$$

where  $Q$  is a symmetric matrix whose entries are determined from the matrix  $A$ , once a special node  $s \in V$  has been selected and a correspondence between the nodes in  $V \setminus \{s\}$  and the entries of  $x$  has been established (see [23]). A feasible solution to (28.17), i.e., an arbitrary binary  $n - 1$  vector, is readily mapped to a cut  $\delta(S)$  of  $G$ , by defining  $S$  to be the set containing  $s$  and the nodes whose associated  $x$  entry is zero.

Due to the very simple structure of (28.17), it is relatively easy to design a heuristic algorithm for this problem and, as a matter of fact, many such algorithms, based on a variety of methods (evolutionary algorithms, simulated annealing, tabu-search, local search, etc.) have been proposed in the literature. One of the most recent references is, e.g., [6] which also contains pointers to many other papers.

In this section we only consider some heuristic algorithms that strongly rely on the solution of the SDP relaxation of Max-Cut.

One of the papers that have mainly contributed to making SDP techniques popular, in particular in the field of the approximation of hard combinatorial problems, is the one by Goemans and Williamson [15]. The paper describes and analyzes a randomized algorithm for Max-Cut that can be briefly outlined as follows.

### Goemans–Williamson Algorithm

**Data:**  $C$

**S.1** Solve (28.1) and obtain an optimal solution  $X$  and the optimal value  $z_P$ .

**S.2** Apply the Cholesky decomposition to the matrix  $X$  and obtain a matrix  $V \in \mathbb{R}^{n \times r}$  such that  $X = VV^T$ . Let  $v_1, v_2, \dots, v_n$  be the row vectors of  $V$ .

**S.3** From a uniform distribution, draw the components of a vector  $h \in \mathbb{R}^r$ .

**Return**  $z_P$  and the cut  $\delta(S)$  where  $S = \{i : \langle h, v_i \rangle \geq 0\}$ .

We assume here that all the components of the weighted adjacency matrix of  $G$  are non-negative. The expected weight  $\bar{W}$  of the cut is given by  $\bar{W} = \sum_{ij \in \delta(S)} A_{ij} p_{ij}$ , where  $p_{ij}$  is the probability that  $ij$  belongs to the cut or, equivalently, the probability that  $v_i$  and  $v_j$  lay on opposite sides with respect to the hyperplane defined by  $\langle h, x \rangle = 0$ . Such a probability is proportional to the angle defined by the two vectors. Finally, using the inequality  $\arccos(\alpha)/\pi \geq 0.87856(1 - \alpha)/2$ , we can write

$$\bar{W} = \sum_{ij} A_{ij} \frac{\arccos(\langle v_i, v_j \rangle)}{\pi} \geq 0.87856 \sum_{ij} A_{ij} \frac{1 - \langle v_i, v_j \rangle}{2} = 0.87856 z_P.$$

In conclusion, the expected gap (in percentage with respect to the SDP bound) obtained by using the relaxation (28.1) and the Goemans–Williamson algorithm is around 12.1%.

Once the SDP bound has been computed, the main computational effort of the Goemans–Williamson algorithm, is essentially devoted to finding the vectors  $v_i$ , with  $i = 1, \dots, n$ . This task can be accomplished by a truncated Cholesky decomposition of the matrix  $X$  which requires  $O(n^3)$  operations (so that computational time is proportional to  $n^3$ ) and needs memory space proportional to  $n^2$ . Therefore, the algorithm cannot be applied to very large instances with size, say, on the order of one hundred thousand nodes.

By exploiting a particular implementation of the IRA scheme described in Sect. 28.3.3, in [17] an efficient heuristic for finding good solutions for very large graphs has been proposed. Indeed, an efficient implementation of the IRA scheme makes it possible to apply the Goemans and Williamson approximation algorithm to very large graphs, since on the one hand it is able to solve problem (28.10) in reasonable time also for very large graphs, and on the other hand, it outputs (for free) the vectors  $v_i$ , avoiding the need of a Cholesky factorization. The cut provided by the Goemans and Williamson algorithm is then improved by means of a 1-opt local search, where all possible moves of a single vertex to the opposite set of the partition are checked. In [12], a particularly successful step is to re-apply the heuristic starting from a convex combination of the solution  $X$  of the relaxation and of the rank one matrix  $\hat{x}\hat{x}^T$ , where  $\hat{x}$  is the current best cut. This procedure aims at moving towards a “good vertex”. If the graph is too large, however, this is not practical since it requires

the Cholesky factorization of the matrix  $\alpha X + (1 - \alpha)\hat{x}\hat{x}^T$ . In order to move towards a “good vertex” without using any Cholesky factorization, in [17] a perturbation  $C'$  of the original objective function matrix  $C$  in (28.1) is applied, that is given by  $C' = C - \beta\hat{x}\hat{x}^T$  with  $\beta > 0$ . Such a perturbation has the same effect of moving toward a “good vertex” as by increasing  $\beta$ , the weights of the edges in the cut defined by  $\hat{x}$  get increased, while the weights of those outside of this cut get decreased. The SDP problem (28.1) with  $C = C'$  is then solved by again using the IRA scheme and the whole heuristic is repeated a given number of times. Summarizing, the scheme of the heuristic algorithm, proposed in [17], is the following:

**IRA-based Heuristic**

**Data:**  $C, \gamma > 0, k_{\max}$ .

**Initialization:** Set  $\bar{Q} = \sum_{i,j} |C_{ij}| / |E|$ ,  $\hat{x} = e$ .

**For**  $k = 0, \dots, k_{\max}$ :

- S.0** Set  $\beta^k = (k_{\max} - k)\gamma\bar{Q}$
- S.1** Apply IRA to problem (28.10) with  $Q = -C + \beta^k\hat{x}\hat{x}^T$ , and let  $v_i, i = 1, \dots, n$ , be the returned solution and  $-z_{LB}$  a valid bound for the corresponding Max-Cut problem.
- S.2** Apply the Goemans–Williamson hyperplane rounding technique to the vectors  $v_i, i = 1, \dots, n$ . This gives a bipartition representative vector  $\bar{x}$ .
- S.3** Locally improve the cut  $\bar{x}$  by checking all possible moves of a single vertex to the opposite set of the bipartition. This gives a new bipartition representative vector  $\tilde{x}$ .  
If  $\langle C, \tilde{x}\tilde{x}^T \rangle > \langle C, \hat{x}\hat{x}^T \rangle$ , set  $\hat{x} = \tilde{x}$ .

**End**

**Return** Best cut  $\hat{x}$ , lower bound  $\langle C, \hat{x}\hat{x}^T \rangle$ , upper bound  $-z_{LB}$ .

Note that the perturbation decreases when the iteration counter increases, getting to zero in the last iteration. On the other hand, in the first iteration the whole matrix is perturbed. We stress that, after the first iteration, Step S.1 is not expensive since a warm start technique can be used: at each iteration IRA is started by the solution found at the previous step, so that the new minimization is computationally cheap. Numerical results for this heuristic are reported in Sect. 28.6. Another advantage of this heuristic is that it also provides a performance measure, since it computes both an upper and lower bound on the value of the optimal cut.

## 28.5 How to Find Optimal Max-Cut Solutions

As mentioned in Sect. 28.1, Max-Cut is an NP-hard problem and requires using some pseudo-enumeration techniques in order to be solved to optimality. In a typical solution algorithm, an instance of the problem is recursively replaced by two other

instances obtained by imposing that two nodes of the graph belong to the same set or to two distinct sets of the bipartition, respectively.

Given a graph  $G = (V, E)$  with weighted adjacency matrix  $A$  and a selected node  $s$ , by  $\langle A, U, W \rangle$  we denote the Max-Cut instance defined on  $G$  but with the additional constraints that the nodes of the set  $U \subseteq V$  have to be in the same set of the bipartition as  $s$  and those of a set  $W \subseteq V$  have to be in the other set.

It is interesting to note that  $\langle A, U, W \rangle$  is the instance of a Max-Cut problem in a smaller graph with no additional constraints. This is easily seen from the formulation (28.17) of the problem. Consider the two instances that are obtained from a given one by imposing that a variable  $x_i$  has to be equal to 0 or to 1, respectively. We obtain two problems again of the form (28.17) with one less variables and with matrices  $Q_0$  and  $Q_1$ , respectively. Matrix  $Q_0$  is obtained from  $Q$  by deleting the  $i$ -th row and column, while  $Q_1$  is obtained from  $Q_0$  by suitably modifying its diagonal. In the latter case a constant has also to be added to the objective function as well.

We call  $M(\langle A, U, W \rangle)$  the weighted adjacency matrix of the graph that corresponds to the instance  $\langle A, U, W \rangle$  and  $k(\langle A, U, W \rangle)$  the constant that has to be added to the weight of a cut in this graph to obtain the weight of the corresponding cut in  $G$ . The size of  $M(\langle A, U, W \rangle)$  is  $|V| - |U| - |W| + 1$ , thus the larger the sets  $U$  and  $W$ , the smaller (and simpler) is the corresponding Max-Cut instance.

The process of replacing a problem by two simpler subproblems can be represented by a binary tree. The growing of the tree from a node is interrupted as soon as the *upper bound*, i.e., the value of a bound on the Max-Cut optimal value for the instances corresponding to that node (found, e.g., by solving (28.1)) falls below the *lower bound*, i.e., the weight of the best available cut (found, e.g., with the heuristic techniques of Sect. 28.4). If this never happens, the procedure produces a tree of  $2^{n-1}$  nodes which amounts to solving the problem by complete enumeration.

The above scheme is computationally acceptable only if the number of nodes is kept relatively small, which is obtained by providing good heuristics and good relaxations.

### 28.5.1 LP Relaxations

A popular and successful relaxation for Max-Cut is based on Linear Programming techniques. An alternative way to encode a cut is by a  $0 - 1$  vector in  $\mathbb{R}^E$  with component, corresponding to the edge connecting the nodes  $i$  and  $j$ , equal to 1 if the edge  $ij$  belongs to the cut and 0 otherwise. The convex hull of the encoding vectors of all cuts of  $G$  is the Cut Polytope  $C(G)$ . Then solving Max-Cut is equivalent to optimizing a linear function over  $C(G)$ . Due to the enormous number of inequalities that are needed to define this polytope (most of which are unknown), one has to use simpler and larger polytopes.

A polytope containing  $C(G)$ , that can thus be used to obtain a relaxation of Max-Cut, is the Semimetric Polytope  $\mathcal{M}(G)$  which is defined by the following inequalities

$$\begin{aligned} \sum_{ij \in F} z_{ij} - \sum_{ij \in C \setminus F} z_{ij} &\leq |F| - 1 \text{ for all cycles } C \text{ of } G \text{ and for} \\ &\quad \text{all } F \subseteq C \text{ with } |F| \text{ odd} \\ 0 \leq z_{ij} \leq 1 & \quad \text{for all } ij \in E \end{aligned} \tag{28.18}$$

that encode the property that the intersection of a cut with a cycle of  $G$  always has an even cardinality. For this reason the inequalities in the first set of (28.18) are called the *cycle inequalities*.

In general the number of cycle inequalities is exponential in  $n$ . However, optimizing a linear function over  $\mathcal{M}(G)$  is still efficiently doable with a *cutting plane algorithm*. One starts by optimizing a linear program defined by a very small subset of (28.18) and then checks if the optimal solution satisfies the whole set. In the affirmative case the algorithm stops, otherwise it adds some violated inequalities to the formulations and iterates the process. Despite the exponential size of the set of cycle inequalities, solving the *separation problem* for the cycle inequalities, i.e., finding one which is violated by a given point, can be done in polynomial time [3]. This fact, combined with the efficiency of today's linear program solvers, makes one expect a fast computation of the semimetric bound. Unfortunately, this is true only for very sparse graphs (like grid graphs where the number of edges is only twice the number of nodes). For these graphs using the semimetric bound makes it possible to optimize instances of several thousand nodes in a reasonable amount of time (see, e.g., [33]).

But what about denser instances? In this case a stronger relaxation would be necessary which could be obtained by enriching the set of the valid inequalities that are used in the above cutting plane algorithm. Several families of valid or even facet defining inequalities for  $C(G)$  have been characterized and thus provide a source for possible stronger relaxations.

Observe also that a cycle inequality is not facet defining, and thus is not essential in the formulation of a relaxation, if the cycle contains a chord. In addition the trivial inequalities  $0 \leq z_{ij} \leq 1$  are not facet defining if the edge  $ij$  is contained in a triangle. Therefore, if  $G$  is a complete graph with  $n$  nodes, denoted by  $K_n$ , the corresponding Semimetric Polytope  $\mathcal{M}(K_n)$  is completely described by cycle inequalities where each cycle has length 3. Such inequalities are also called the *triangle inequalities*.

A detailed treatment on most of the families of facet defining inequalities for  $C(G)$  known to date is contained in the book [10]. Unfortunately, the separation problem is difficult for most of them. In addition, many such inequalities are defined only for the cut polytope on a complete graph and it is not easy to exploit them when the graph is not complete. Because of these difficulties, no computational studies are available today, where the use of an enriched set of inequalities is the key for solving difficult dense instances.

### 28.5.2 SDP Relaxations

A breakthrough in the exact solution of dense instances was made when the SDP relaxation was introduced: instances on complete graphs of 50 to 100 nodes, very difficult for Linear Programming techniques, could be solved in very reasonable amount of time by exploiting the SDP bound in the above mentioned pseudo-enumeration scheme.

Unfortunately, the bound provided by (28.1) is not particularly strong. For example, using that bound, an instance of 100 nodes can be solved in about 20 min on a laptop after the generation of 30,000 nodes. These numbers tend to increase very rapidly as  $n$  increases. Therefore, instances of slightly larger size remain practically unsolvable if one only relies on the bound given by (28.1).

To generate stronger bounds several formulations have been proposed, building on (28.1) by enlarging (even substantially) the dimension of the solution space (see e.g. [2, 31]). However, these formulations do not appear, at the moment, to be of practical use for instances of 100 or more nodes.

We said that for denser graphs the SDP bound outperforms the semimetric bound in terms of computation time. But what about the qualities of the two bounds? Is there one of the two that is always preferable to the other? The answer is unfortunately no: either one can be the winner, depending on the weighted adjacency matrix of the graph.

The projection of the feasible set of (28.1) onto the space of the off-diagonal and upper-triangular components of  $X$  is a convex body, called *elliptope* or Max-Cut *spectrahedron* (see e.g. [32]), that contains the cut polytope  $C(K_n)$ . Therefore, a very natural way to improve the SDP bound would be to optimize over the intersection of the elliptope with the semimetric polytope or with any stronger relaxation obtained by adding inequalities, valid for  $C(K_n)$ , to the triangle inequalities.

### 28.5.3 Combining LP and SDP Relaxations

Two ways of optimizing over this intersection have been experimented. The first is described in [24], where, at every iteration of the bound computation, the triangle inequalities are added to the formulation (28.1), like it is done in the linear programming based cutting plane algorithms. The resulting SDP problem is solved with an interior-point algorithm. The bound computation is then embedded into a pseudo-enumeration scheme. The resulting algorithm is able to solve instances up to 100 nodes. However, the SDP computation becomes slower and slower as long as new inequalities are added. Therefore, the approach becomes computationally impractical for graphs of more than 100 nodes.

A different approach is adopted in [40] where a branch and bound algorithm, named BiqMac, is described. The bound is computed iteratively by adding triangle inequalities to the formulation as it is done in cutting plane algorithms. However,

**Data:** A complete graph  $G = (V, E)$ ,  $A$  weighted adjacency matrix,  $s \in V$ .  
**Initialization:** Set the problem list  $\mathcal{P} = \{\langle A, \{s\}, \emptyset \rangle\}$ ,  $lb = 0$ , and  $S = \emptyset$ .  
**While**  $\mathcal{P} \neq \emptyset$  **do**

- S.0** Extract  $\langle A, U, W \rangle$  from  $\mathcal{P}$ .
- S.1** Compute the Laplacian matrix  $L$  associated with the matrix  $M(\langle A, U, W \rangle)$  and set  $C = \emptyset$ .
- S.2 While** Condition (A) is satisfied **do**

  - S.2.0** Define the Lagrangian

$$\mathcal{L}(X, \gamma) = \langle L, X \rangle + \sum_{C \in C} \gamma(C)r(B) - \sum_{C \in C} \gamma(C)\langle C, X \rangle.$$
- S.2.1** Define the dual functional
$$f(\gamma) = \max\{\mathcal{L}(X, \gamma) : \text{diag}(X) = e, X \succeq 0\}.$$
- S.2.2** Solve
$$z_{\text{mc-met}} = \min\{f(\gamma) : \gamma \geq 0\}$$

by the bundle method, using SDP optimization to evaluate  $f(\gamma)$ . Let  $\tilde{X}$  be the optimal solution.

- S.2.3** Solve the separation problem for the triangle inequalities relative to the point  $\tilde{X}$ . If any inequalities are found, for each of them add the corresponding matrix  $C$  to the set  $C$ .
- end**
- S.3** Using the last solution  $\tilde{X}$  computed, find a ‘good’ cut using the Goemans-Williamson algorithm, possibly followed by some local improvement procedure. Let this cut be defined by the set  $\bar{S}$ , with  $s \in \bar{S}$  and let  $\bar{z}$  be its value.  
If  $\bar{z} + k(\langle A, U, W \rangle) > lb$ , then set  $lb = \bar{z} + k(\langle A, U, W \rangle)$  and  $S = \bar{S} \cup U$ .
- S.4** If  $z_{\text{mc-met}} + k(\langle A, U, W \rangle) > lb$ , then select a node  $t \in V \setminus (U \cup W \cup \{s\})$  and add the two instances  $\langle A, U \cup \{t\}, W \rangle$  and  $\langle A, U, W \cup \{t\} \rangle$  to  $\mathcal{P}$ .

**end**  
**Return**  $S$  and  $lb$ .

**Fig. 28.1** The BiqMac Algorithm

differently from the previous method, the feasible set of (28.1) is left unchanged, while the objective function is modified at every iteration by dualizing all the identified inequalities in a Lagrangian fashion. The dual functional is then optimized using the bundle method.

The algorithm can be outlined as in Fig. 28.1, where  $C$  denotes the weighted adjacency matrix of a graph with node set  $V$ , edge set  $E$  and weights given by a triangle inequality,  $r(C)$  denotes its right hand side and  $\gamma(C)$  the dual variable associated with the inequality.

Two technical details in the algorithm described in Fig. 28.1 need to be clarified. The first concerns Condition (A) in Step S.2. This condition is satisfied as long as the separation problem finds some triangle inequalities violated and as long the process of adding more inequalities to the set  $C$  produces a satisfactory decrease in

the value  $z_{\text{mc-met}}$ . Observe that, as the triangle inequalities are  $4\binom{n}{3}$ , the separation procedure trivially amounts to checking each of them for violation. The other detail concerns the way node  $t$  is chosen in Step S.4. There are several rules, the easiest being the one where the chosen node corresponds to the entry of  $\tilde{X}$  with smallest absolute value in the row of node  $s$ .

## 28.6 Computational Status

In this section, we explore the actual computational status for solving the SDP problem (28.1), and for finding good cuts for the Max-Cut problem (28.3). Furthermore, we report a summary of the computational behavior of the BiqMac algorithm.

As an interior-point method we select the dual-scaling algorithm defined in [5] and implemented in the software DSDP. DSDP is considered particularly efficient for solving problems where the solution is known to have low rank (as it is the case for Max-Cut instances), since it exploits low-rank structure and sparsity in the data. Furthermore, DSDP has relatively low memory requirements for an interior-point method, and is indeed able to solve instances up to 10,000 nodes. We also include the spectral bundle method SB of [25] described in Sect. 28.3.2.

Among the non-linear programming based methods, we consider the different approaches described in Sect. 28.3.3 for solving problem (28.11) as required at Step S.1 of the IRA scheme. As mentioned in Sect. 28.3.3, in [8] two different unconstrained formulations have been proposed for the solution of the non-linear problem (28.11) and have been implemented in two C codes. The first one is a general purpose code for solving linear SDPs based on the sequential Augmented Lagrangian methods with the minimization performed by means of a limited memory L-BFGS method. The corresponding IRA algorithm is implemented in the code SDPLR which does not include explicitly Step S.2. The IRA scheme for the special structured problem (28.10) based on the second unconstrained formulation is implemented in the code SDPLR-MC. This code is based on the unconstrained formulation (28.15) solved by means of an L-BFGS algorithm, with the updating rule for the values of the rank  $r^j = \lceil \frac{j}{p} \tilde{r} \rceil$  ( $\tilde{r}$  given by (28.12) and  $p = 5$ ). The computational results in [8] show that the method SDPLR-MC outperforms SDPLR, thus we include in our comparison only SDPLR-MC. We remark again that both SDPLR and SDPLR-MC do not certify global optimality of the produced solution for problem (28.1) in the sense that they do not check the global optimality condition  $Q + \text{Diag}(\lambda(\hat{V})) \succeq 0$  at Step S.2 of the IRA scheme.

The manifold optimization method GenRTR defined in [29] is implemented in Matlab, so that direct comparison in terms of computational time with other C, C++ or Fortran codes is not really fair. However, in [29] the authors report a comparison on some Max-Cut instances of GenRTR and SDPLR. The reported results show that the two methods may be considered to have comparable performances. This implies

**Table 28.1** CPU time of the three NLP based methods, SB and DSDP

graph	$ V $	$ E $	SpeeDP	EXPA	SDPLR-MC	SB	DSDP
1	250	1,283	0.04	0.15	0.14	0.45	0.52
2	500	2,355	0.11	0.52	0.43	0.98	2.40
3	1,000	5,909	0.24	3.07	3.10	6.74	15.70
4	2,000	11,778	1.45	5.03	10.22	31.13	117.00
5	5,000	29,570	15.74	77.94	45.68	368.23	1969.00
6	10,000	20,000	8.21	35.77	32.92	1471.95	6017.00

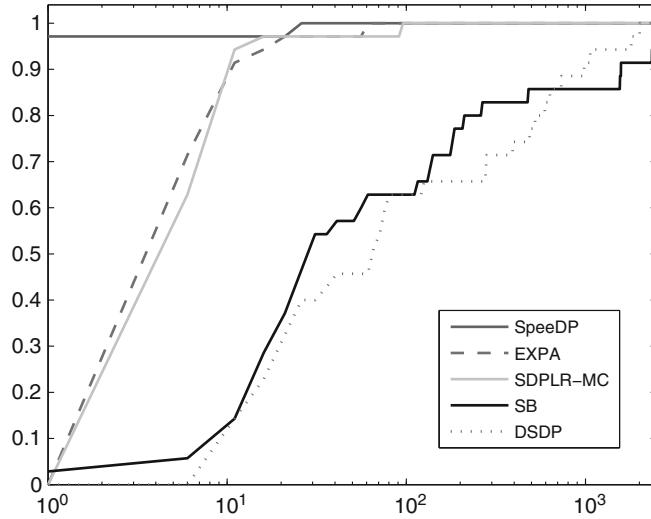
that on the special structured problem (28.1) that comes from Max-Cut, GenRTR should have worse performances than SDPLR-MC, so that we do not include GenRTR in our comparison.

The IRA algorithm based on the exact penalty approach proposed in [19] is implemented in the code EXPA where the minimization of the exact penalty  $P(V)$  is performed by a non-monotone Barzilai–Borwein gradient method [20]. In [16] this same method is used for solving the unconstrained problem (28.16). The corresponding IRA algorithm is implemented in the code SpeeDP with the updating rule of the rank  $r^{j+1} = \min\{1.5 \times r^j, \hat{r}\}$   $j = 1, 2, \dots$ , with a dimension based heuristic for the choice of the value  $\hat{r}$ . We remark that both EXPA and SpeeDP include the evaluation of the minimum eigenvalue at Step S.2 of the IRA scheme using the ARPACK subroutines *dsaupd* and *dseupd*.

Thus, to summarize, we report comparison among DSDP, SB, SDPLR-MC, EXPA and SpeeDP as done in [17] where an extensive numerical comparison of the performance of the different codes on a test bed consisting of 38 widely used instances of Max-Cut is presented. To give a flavor of the behavior of the methods, we report here only the results on six selected problems with an increasing dimension ranging from  $|V| = 250$  to  $|V| = 10,000$  and from  $|E| \approx 1,300$  to  $|E| \approx 30,000$  (*mcp250-3*, *mcp500-3*, *G51* from the SDPLIB collection <http://euler.nmt.edu/~brian/sdplib/sdplib.html> and *G35*, *G58*, *G72* from the Gset collection <http://dollar.biz.uiowa.edu/~sburer/SDPLR/problems/maxcut/>).

We compare the performance of the different codes in term of computational time and accuracy. The CPU time, in seconds, on each instance are reported in Table 28.1 together with the dimension  $|V| = n$  and  $|E|$ .

It emerges that the time increases with the dimension, as expected. The non-linear programming based methods are faster than both the interior-point and spectral bundle methods. Among the non-linear programming based methods, SpeeDP shows the most competitive performance. This is true not only on these six problems but on the overall test bed of 38 problems. Complete tables related to these numerical tests can be found in [17]. To give a view of the behavior with respect to the computational time on the whole test bed (except the two largest problems *G77* and *G81* of the Gset collection where DSDP runs out of memory), we draw the performance profile [11] in Fig. 28.2. We recall that the higher a method’s profiles appears, the better is the method’s performance.



**Fig. 28.2** Comparison between NLP based methods, SB and DSDP (the x-axis is the CPU time in seconds reported in logarithmic scale)

As for the accuracy, a comparison among primal and/or dual objective function values obtained by the five methods is required. We first observe that in the non-linear programming methods falling within the IRA scheme (SDPLR-MC, EXPA, SpeeDP), the primal objective value  $\langle Q, \widehat{V}\widehat{V}^T \rangle$  is always reported on exit together with a measure of dual infeasibility  $\lambda_{\min}(Q + \Lambda(\widehat{V}))$ . Whenever  $\lambda_{\min}(Q + \Lambda(\widehat{V})) \geq 0$  perfect dual feasibility is obtained, so that the value of the primal function gives a valid bound for the Max-Cut problem (28.3). Whenever this is not true a valid bound, namely the value of the dual function, is obtained as  $\langle Q, \widehat{V}\widehat{V}^T \rangle + n \min\{0, \lambda_{\min}(Q + \Lambda(\widehat{V}))\}$ . We note that the codes EXPA and SpeeDP give this value as an output, whereas SDPLR-MC only reports the primal objective value. Interior-point methods report both primal and dual values, while the spectral bundle method SB produces a value of the dual objective function that is a bound on the optimal value of problem (28.1).

By analyzing the results on the whole test bed (see tables and figures in [17]), it emerges that non-linear programming methods and SB methods have usually worse accuracy than interior-point methods. SpeeDP is an exception, since its accuracy can be considered comparable with interior-point methods. Indeed SpeeDP finds perfect dual feasibility on 20 problems, it is more accurate than DSDP on eight problems (comparing the dual objective function) and, on the ten problems where SpeeDP is less accurate, the difference is less than  $10^{-4}$ . This comparable level of accuracy is obtained with a significant saving of time when the dimension of the problem increases.

Now, we consider the state-of-the-art method for solving the Max-Cut problem to optimality, the BiqMac algorithm described in Sect. 28.5.3. BiqMac solves any instance of size up to  $n = 100$  routinely and larger instances of a special structure up

**Table 28.2** 6-regular graphs with 1,030,301 nodes and 3,090,903 edges

Weights	Total CPU time	Upper bound	Best cut	gap%
1	4,723	3,090,133	3,060,300	0.97
[1, 10]	22,042	15,454,739	15,338,007	0.76
[1, 1000]	29,072	1,545,550,679	1,534,441,294	0.72
[-100, 100]	47,491	57,288,795	49,111,079	14.27

to  $n = 300$ . An extensive computational test can be found in [40], where the solution of some still unsolved instances is reported. A server based on this algorithm, to which one can submit Max-Cut as well as unconstrained binary quadratic instances is accessible at the site [39] or via the NEOS Server for Optimization.

Finally, we recall some numerical results obtained by a heuristic that fits within the scheme described in Sect. 28.4. Indeed, in [17] the heuristic SpeeDP-MC, based on the use of the SpeeDP algorithm within the scheme of Sect. 28.4, has been proposed. In [17] SpeeDP-MC has been tested on large random graphs, with number of nodes between  $n = 500$  and  $n = 2,500$  for edge densities varying between 10% and 100% and weights drawn from different distributions. It emerges from the tests that the heuristic is able to produce a good cut in a small amount of time and, as expected, the performance of the heuristic is better on sparse graphs in term of time, but the gap (intended as the distance between the solution found and the bound produced) decreases when the density of the graph increases since the quality of the bound improves. We consider it worth mentioning here that in [17] SpeeDP-MC has been used to compute good feasible solutions of the Max-Cut problem on some huge 6-regular random graphs (3D toroidal grid graphs) with millions of nodes and edges and different weight ranges. These results, reported in Table 28.2, were surprisingly good.

Summarizing, we believe that the efficiency of non-linear programming based methods may be successfully used within heuristic and/or exact schemes for solving Max-Cut. Methods in other classes (such as interior-point or spectral bundle methods) turn out to be extremely accurate and more naturally extendable to classes of SDP problems more general than problem (28.1).

## 28.7 Conclusions

In this chapter, we surveyed some recent results on the Max-Cut problem, focusing mainly on the impact that the progress in SDP development has on the computation of good/optimal solutions of this problem. Since the Max-Cut problem is NP-hard, some pseudo-enumeration technique is needed to solve it exactly. Therefore the computation of good upper and lower bounds is fundamental for any solution algorithm for the Max-Cut problem, and the last years of research showed that the simple SDP problem (28.10) plays a fundamental role in all the most efficient

algorithms. This SDP problem is also contained in the relaxations arising from some related problems: Max- $k$ -Cut, Coloring and Ordering problems described in Sect. 28.2. In the chapter we surveyed the methods available for solving this SDP problem: interior-point methods, the spectral bundle method and a recent class of non-linear programming based methods, that have a huge computational impact on the ability of solving large sparse instances. A method belonging to this class has been exploited to define a heuristic algorithm which is able to find good solutions for huge instances of Max-Cut as described in Sect. 28.4. Furthermore, we described BiqMac, the best SDP based method available in the literature for solving the Max-Cut problem to optimality. We concluded the chapter by a computational section where all the methods described are compared and where a summary of the numerical results shows the big impact of the new non-linear programming based methods on the size of the instances of Max-Cut that can be tackled.

**Acknowledgements** We thank the two anonymous referees for their careful reading of the chapter, and for their constructive remarks that greatly helped to improve its presentation.

## References

1. Absil, P.-A., Baker, C.G., Gallivan, K.A.: Trust-region methods on Riemannian manifolds. *Journal Foundations of Computational Mathematics* **7**(3), 303–330 (2007)
2. Anjos, M.F., Wolkowicz, H.: Strengthened semidefinite relaxations via a second lifting for the Max-Cut problem. *Discrete Applied Mathematics* **119**(1-2), 79–106 (2002)
3. Barahona, F., Mahjoub, A.R.: On the cut polytope. *Mathematical Programming* **36**(2), 157–173 (1986)
4. Barvinok, A.: Problems of distance geometry and convex properties of quadratic maps. *Discrete Computational Geometry* **13**, 189–202 (1995)
5. Benson, S.J., Ye, Y., Zhang, X.: Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM Journal on Optimization* **10**(2), 443–461 (2000)
6. Boros, E., Hammer, P.L., Tavares, G.: Local search heuristics for quadratic unconstrained binary optimization. *Journal of Heuristics* **13**, 99–132 (2007)
7. Buchheim, C., Wiegele, A., Zheng, L.: Exact algorithms for the Quadratic Linear Ordering Problem. *INFORMS Journal on Computing* **22**(1), 168–177 (2010)
8. Burer, S., Monteiro, R.D.C.: A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming* **95**(2), 329–357 (2003)
9. de Klerk, E., Pasechnik, D.V., Warners, J.P.: On approximate graph colouring and Max- $k$ -Cut algorithms based on the  $\vartheta$ -function. *Journal of Combinatorial Optimization* **8**(3), 267–294 (2004)
10. Deza, M., Laurent, M.: *Geometry of Cuts and Metrics*, vol. 15 of Algorithms and Combinatorics. Springer-Verlag, Berlin (1997)
11. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profile. *Mathematical Programming* **91**(2), 201–213 (2001)
12. Fischer, I., Gruber, G., Rendl, F., Sotirov, R.: Computational experience with a bundle approach for semidefinite cutting plane relaxations of Max-Cut and equipartition. *Mathematical Programming* **105**(2-3, Ser. B), 451–469 (2006)
13. Frieze, A., Jerrum, M.: Improved approximation algorithms for Max  $k$ -Cut and Max Bisection. *Algorithmica* **18**(1), 67–81 (1997)

14. Ghaddar, B., Anjos, M.F., Liers, F.: A branch-and-cut algorithm based on semidefinite programming for the minimum  $k$ -partition problem. *Annals of Operations Research*, **188**(1), 155–174 (2011)
15. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the Association for Computing Machinery* **42**(6), 1115–1145 (1995)
16. Grippo, L., Palagi, L., Piacentini, M., Piccialli, V.: An unconstrained approach for solving low rank SDP relaxations of  $\{-1, 1\}$  quadratic problems. Technical Report 1.13, Dip. di Informatica e sistematica A. Ruberti, Sapienza Università di Roma (2009)
17. Grippo, L., Palagi, L., Piacentini, M., Piccialli, V., Rinaldi, G.: SpeeDP: A new algorithm to compute the SDP relaxations of Max-Cut for very large graphs. Technical Report 13.10, DII-UTOVRM - Università di Roma Tor Vergata (2010) available on Optimization Online.
18. Grippo, L., Palagi, L., Piccialli, V.: Necessary and sufficient global optimality conditions for NLP reformulations of linear SDP problems. *Journal of Global Optimization* **44**(3), 339–348 (2009)
19. Grippo, L., Palagi, L., Piccialli, V.: An unconstrained minimization method for solving low rank SDP relaxations of the Max Cut problem. *Mathematical Programming* **126**(1), 119–146 (2011)
20. Grippo, L., Sciandrone, M.: Nonmonotone globalization techniques for the Barzilai-Borwein gradient method. *Computational Optimization and Applications* **23**, 143–169 (2002)
21. Grone, R., Pierce, S., Watkins, W.: Extremal Correlation Matrices. *Linear Algebra Application* **134**, 63–70 (1990)
22. Grötschel, M., Jünger, M., Reinelt, G.: Facets of the linear ordering polytope. *Mathematical Programming* **33**, 43–60 (1985)
23. Hammer, P.: Some network flow problems solved with pseudo-boolean programming. *Operations Research* **13**, 388–399 (1965)
24. Helmberg, C., Rendl, F.: Solving quadratic (0,1)-problems by semidefinite programs and cutting planes. *Mathematical Programming* **82**(3), 291–315 (1998)
25. Helmberg, C., Rendl, F.: A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization* **10**, 673–696 (2000)
26. Hestenes, M.: Multiplier and gradient methods. *Journal of Optimization Theory and Application* **4**, 303–320 (1969)
27. Homer, S., Peinado, M.: Design and performance of parallel and distributed approximation algorithm for the Maxcut. *Journal of Parallel and Distributed Computing* **46**, 48–61 (1997)
28. Hungerländer, P., Rendl, F.: Semidefinite relaxations of ordering problems. Accepted for Publication in *Mathematical Programming B*
29. Journée, M., Bach, F., Absil, P.A., Sepulchre, R.: Low-rank optimization for semidefinite convex problems. *SIAM Journal on Optimization* **20**(5), 2327–2351 (2010)
30. Karger, D., Motwani, R., Sudan, M.: Approximate graph colouring by semidefinite programming. *Journal of the Association for Computing Machinery* **45**, 246–265 (1998)
31. Lasserre, J.B.: An explicit equivalent positive semidefinite program for nonlinear 0-1 programs. *SIAM Journal on Optimization* **12**(3), 756–769 (2002)
32. Laurent, M., Rendl, F.: Semidefinite programming and integer programming. In: Aardal, K., Nemhauser, G.L., Weismantel, R. (eds.) *Handbook in OR & MS, Discrete Optimization*, vol. 12, Chap. 8. Elsevier B.V. (2005)
33. Liers, F., Jünger, M., Reinelt, G., Rinaldi, G.: Computing exact ground states of hard Ising spin glass problems by branch-and-cut. In: Hartmann, A.K., Rieger, H. (eds.) *New Optimization Algorithms in Physics*, pp. 47–69. Wiley-VCH Verlag (2004)
34. Lovász, L.: On the Shannon capacity of a graph. *IEEE Transactions on Information Theory* **25**(1), 1–7 (1979)
35. Pataki, G.: On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of Operations Research* **23**, 339–358 (1998)
36. Poljak, S., Rendl, F., Wolkowicz, H.: A recipe for semidefinite relaxation for 0-1 quadratic programming. *Journal of Global Optimization* **7**, 51–73 (1995)

37. Powell, M.J.D.: A method for nonlinear constraints in minimization problem. In: Optimization, pp. 283–298. Academic Press, New York (1969)
38. Reinelt, G.: The Linear Ordering Problem: Algorithms and Applications. Heldermann Verlag (1985)
39. Rendl, F., Rinaldi, G., Wiegele, A.: Biq Mac Solver - BIInary Quadratic and MAx-Cut Solver. <http://biqmac.uni-klu.ac.at/>.
40. Rendl, F., Rinaldi, G., Wiegele, A.: Solving Max-Cut to optimality by intersecting semidefinite and polyhedral relaxations. Mathematical Programming **121**(2), 307–335 (2010)

# Chapter 29

## Global Approaches for Facility Layout and VLSI Floorplanning

Miguel F. Anjos and Frauke Liers

### 29.1 Introduction

The facility layout problem consists of partitioning a rectangular facility of known dimensions into departments with a given (fixed) area so as to minimize the total cost associated with the (known or projected) interactions between these departments. This cost is modeled as the weighted sum of the center-to-center distances between all pairs of facilities, where the pairwise weights reflect transportation costs and/or adjacency preferences between departments. If the height and width of the departments vary, then finding their optimal (rectangular) shapes is also a part of the problem. This is a hard problem, in particular because any desirable layout must have no overlap among the areas of the different departments. Versions of the facility layout problem occur in many environments, such as flexible manufacturing and service center layout, as well as in other engineering applications, such as the design of Very Large Scale Integration (VLSI) circuits. Nearly all of the resulting problems are known to be NP-hard.

Two types of approaches capable of yielding provably optimal solutions have been proposed in the literature. The first type are graph-theoretic approaches which assume that the desirability of locating each pair of facilities adjacent to each other is known. Initially, the area and shape of the departments are ignored, and each department is simply represented by a node in a graph. Adjacency relationships between departments can now be represented by arcs connecting the corresponding nodes in the graph. The objective is then to construct a graph which maximizes the

---

M.F. Anjos (✉)

Department of Mathematics and Industrial Engineering & GERAD,  
École Polytechnique de Montréal, Montréal, QC, Canada H3C 3A7  
e-mail: [anjos@stanfordalumni.org](mailto:anjos@stanfordalumni.org)

F. Liers

Institut für Informatik, Universität zu Köln, Köln, Germany  
e-mail: [liers@informatik.uni-koeln.de](mailto:liers@informatik.uni-koeln.de)

weight on the adjacencies between nodes. We refer the reader to Foulds [28] for more details. The second type of these approaches are mathematical programming formulations with objective functions based on an appropriately weighted sum of centroid-to-centroid distances between departments. Exact mixed-integer linear programming formulations were proposed in Montreuil [55], Meller et al. [54], and Sheralli et al. [64]. Non-linear programming formulations include the work of Castillo et al. [19, 20]. Most recently, Meller et al. [52] solved problems up to 11 departments to global optimality.

Thus, most of the approaches in the literature that tackle realistically sized problems are based on heuristics with no guarantee of optimality. These include genetic algorithms, tabu search, simulated annealing, fuzzy logic, and many others. Mathematical-programming-based heuristics can also be used to provide high-quality solutions for large problems with 20 or more departments [9, 18]. We refer the reader to the extensive bibliographies in the survey papers of Meller and Gau [53], Mavridou and Pardalos [51], and Singh and Sharma [67].

The survey of Meller and Gau [53] divides the research papers on facility layout into three broad areas. The first is concerned with algorithms for tackling the general layout problem as defined above. The second area is concerned with extensions of the problem in order to account for additional issues which arise in applications, such as designing dynamic layouts by taking time dependency issues into account; designing layouts under uncertainty conditions; and achieving layouts which optimize two or more objectives simultaneously. The third area is concerned with specially structured instances of the problem. One such special case that has been extensively studied occurs when all the facilities have equal dimensions and the possible locations for the departments are given *a priori*; this is the quadratic assignment problem (QAP) formulated by Koopmans and Beckman [43]. Since the possible locations are fixed, the problem reduces to optimize a quadratic objective over all possible assignments of departments to locations. The QAP is NP-hard, and is in general a difficult problem to solve; see Chap. 27 of this Handbook.

In this chapter we will be concerned with two layout problems to which conic optimization approaches have been successfully applied. The first one is a specially structured instance of facility layout, namely the single-row facility layout problem (SRFLP). The SRFLP consists of arranging a given number of rectangular facilities next to each other along a line so as to minimize the total weighted sum of the center-to-center distances between all pairs of facilities. This problem is also known in the literature as the one-dimensional space allocation problem; see, e.g., [58]. The SRFLP also has several interesting connections to other known combinatorial optimization problems such as the maximum-cut problem, the quadratic linear ordering problem, and the linear arrangement problem. We explore some of these connections in Sect. 29.2.3. Note that for the SRFLP, numerous approaches based on linear programming (LP) have been proposed in the literature. Using integer LP methods, the most effective such approach for determining exact solutions is a branch-and-cut algorithm based on the recent model by Amaral [5].

The second problem arises from the application of facility layout to fixed-outline floorplanning in VLSI circuit design. Fixed-outline floorplanning consists

of arranging a set of rectangular modules on a rectangular chip area so that an appropriate measure of performance is optimized, and is hence a specialized version of facility layout. The objective is typically to minimize total wire length.

The impact to date of the conic optimization approaches for facility layout problems, excluding its impact for the QAP addressed elsewhere in this book, can be summarized as follows:

- Anjos and Vannelli [10] used a semidefinite programming (SDP) relaxation to provide globally optimal solutions for SRFLP instances in the literature with up to 30 departments that had remained unsolved for nearly 20 years.
- Hungerländer and Rendl [39] show that the SDP relaxation, when augmented with valid inequalities and optimized using a suitable combination of algorithms, can provide global optimal solutions for SRFLP instances with up to 40 departments. Their approach can solve larger instances to optimality than the approach by Amaral [5].
- Anjos and Yen [11] obtained tight global bounds for SRFLP instances with up to 100 facilities using an alternate SDP relaxation. Their approach consistently achieves optimality gaps not greater than 5%.
- Takouda et al. [69] provided non-trivial lower bounds for the VLSI macro-cell floorplanning problem. Furthermore, their second-order cone programming (SOCP) formulations (described in Sect. 29.3.1) for the area and aspect ratio constraints were key to the success of the two-stage convex optimization-based methodology for floorplanning of Luo et al. [50] and of the facility layout model of Jankovits [40], both of which provide computational results outperforming those previously reported in the literature.

This chapter provides an overview of the SOCP/SDP models underpinning this impact, and concludes with a summary of further research directions.

## 29.2 Introduction to the Single-Row Facility Layout Problem

An instance of the SRFLP is formally defined by  $n$  one-dimensional facilities with given positive lengths  $\ell_1, \dots, \ell_n$ , and pairwise non-negative weights  $c_{ij}$ . The objective is to arrange the facilities so as to minimize the total weighted sum of the center-to-center distances between all pairs of facilities. If all the facilities have the same length and all the non-zero weights are equal, the SRFLP becomes an instance of the linear arrangement problem, see e.g. [47], which is itself a special case of the QAP; see e.g. [21]. (For a survey on the linear arrangement and other graph layout problems, we refer to [27].) Several practical applications of the SRFLP have been identified in the literature, such as the arrangement of departments on one side of a corridor in supermarkets, hospitals, or offices [65], the assignment of disk cylinders to files [58], the assignment of airplanes to gates in an airport terminal [68], and the arrangement of machines in flexible manufacturing systems,

where machines within manufacturing cells are often placed along a straight path travelled by an automated guided vehicle [37]. We refer the reader to the book of Heragu [35] for more information.

Several heuristic algorithms for the SRFLP have also been proposed. We point out the early work of Hall [31], the application of non-linear optimization methods by Heragu and Kusiak [38], the simulated annealing algorithms proposed independently by Romero and Sánchez-Flores [61] and Heragu and Alfa [36], a greedy heuristic algorithm proposed by Kumar et al. [44], and the use of both simulated annealing and tabu search by de Alvarenga et al. [24]. However, these heuristic algorithms do not provide a guarantee of global optimality, or an estimate of the distance from optimality.

Simmons [65] was the first to state and study the SRFLP, and proposed a branch-and-bound algorithm. His subsequent note [66] mentioned the possibility of extending the dynamic programming algorithm of Karp and Held [41] to the SRFLP, which was done by Picard and Queyranne [58]. Different mixed-integer LP models have been proposed [2, 3, 38, 49], and the first polyhedral study of the so-called distance polytope for SRFLP was carried out by Amaral and Letchford [6]. Most recently, the IP-based cutting-plane algorithm of Amaral [5] can compute optimal solutions for instances with up to 35 facilities within a few hours of computing time. The polyhedral structure of Amaral's model has recently been studied by Sanjeevi and Kianfar [63].

At the time of writing, the state-of-the-art in terms of globally optimal methods are the SDP-based approaches. The first SDP relaxation was proposed in [7] where global bounds for instances with up to 80 facilities were obtained using the spectral bundle solver SB [33, 34]. While this solver is able to handle very large SDP problems, a major drawback is that its convergence slows down significantly after several hours of computation. As a consequence, instances with 80 facilities were the largest for which bounds could be obtained in reasonable time in [7], and some of the optimality gaps were greater than 10%. Anjos and Yen [11] obtained bounds for instances with up to 100 facilities using the solver CSDP [14] and an alternate SDP relaxation. Their approach consistently achieves optimality gaps not greater than 5% for even extremely large instances of SRFLP. Most recently, Hungerländer and Rendl [39] were able to obtain globally optimal solutions for instances with up to 40 facilities.

In the following, we will describe LP- and SDP-based approaches for SRFLP in more detail. It turns out that the models used are closely related to an important combinatorial optimization problem called the maximum-cut problem in an undirected graph. The latter asks for partitioning the nodes of a graph into two sets such that the sum of the weights of edges with nodes in different partitions is maximum. First, we introduce binary quadratic optimization problems which are well known to be equivalent to the maximum-cut problem. We then show that the convex hull of solutions feasible for SRFLP is a special instance of a quadratic linear ordering problem which is known to induce a face of a cut polytope. This gives some theoretical evidence that solution approaches for SRFLP based on the

maximum-cut problem can lead to effective algorithms. In fact, we describe how the most effective approaches for solving SRFLP instances exploit the underlying cut-polytope structure.

### 29.2.1 Binary Quadratic Optimization and the Maximum-Cut Problem

Let us consider some combinatorial optimization problem on a finite set  $E$ . We denote the feasible solutions by  $\mathcal{I} \subseteq 2^E$ . The objective function  $c(I) = \sum_{e \in I} c_e$  is linear, where  $c_e \in \mathbf{R}$  for all  $e \in E$ . Without loss of generality, we want to minimize  $c(I)$  over all  $I \in \mathcal{I}$ . Let  $P \subseteq \mathbf{R}^E$  denote a polytope with the property  $x \in \{0, 1\}^E \cap P$  if and only if  $x$  is the characteristic vector of a feasible solution.

The corresponding integer LP problem reads

$$(P) \quad \begin{aligned} & \min \sum_{e \in E} c_e x_e \\ & \text{s.t. } x \in P \\ & \quad x \in \{0, 1\}^E. \end{aligned}$$

In the following, we focus on objective functions that are quadratic in the variables  $x$ , i.e., we consider problems of the form

$$(QP) \quad \begin{aligned} & \min \sum_{e \in E} c_e x_e + \sum_{e, f \in E; e \neq f} c_{ef} x_e x_f \\ & \text{s.t. } x \in P \\ & \quad x \in \{0, 1\}^E. \end{aligned}$$

For problems defined on a graph  $G = (V, E)$  with variables corresponding to edges, and for two edges  $e = \{i, j\}$  and  $f = \{k, l\}$ , we will use the notations  $c_{ef}$  and  $c_{ijkl}$  interchangeably.

In order to linearize (QP) using the standard linearization, we introduce a binary variable  $y_{ef}$  for each pair  $\{e, f\}$  with  $c_{ef} \neq 0$ , modeling  $x_e x_f$ , along with the constraints  $y_{ef} \leq x_e$ ,  $y_{ef} \leq x_f$ , and  $y_{ef} \geq x_e + x_f - 1$ .

The easiest example for (QP) is the quadratic unconstrained binary optimization problem (QUBO), where we optimize in (QP) over the unit hypercube without further constraints. It is well known that QUBO is equivalent to the maximum-cut problem [25, 32]. Given a graph  $G = (V, E)$  with edge weights  $w_e$  and  $W \subseteq V$ , the cut  $\delta(W)$  is defined as

$$\delta(W) = \{\{u, v\} \in E \mid u \in W, v \notin W\}.$$

Its weight is  $\sum_{e \in \delta(W)} w_e$ . The maximum-cut problem asks for a cut of maximum weight and is NP-hard for general graphs. The corresponding cut polytope, i.e., the convex hull of incidence vectors of cuts, is well studied [12, 26], and branch-and-cut implementations that are efficient in practice exist for its solution [46, 60].

In order to establish the equivalence of the maximum-cut problem with QUBO defined on  $m$  binary variables, we construct an auxiliary graph  $G_{\text{lin}} = (V_{\text{lin}}, E_{\text{lin}})$  with  $m + 1$  nodes [25, 32].  $G_{\text{lin}}$  contains a node for each linear variable  $x_e$ . For each non-zero quadratic term  $x_e x_f$  in the objective,  $E_{\text{lin}}$  contains an edge between the nodes corresponding to  $x_e$  and  $x_f$ . Furthermore, an additional root node and edges from this node to all other nodes are introduced. Then there exists a simple linear transformation between the edge variables of  $G_{\text{lin}}$  in the maximum-cut setting and the linear variables and their products in the unconstrained quadratic optimization setting. Under this transformation,  $P$  is isomorphic to the cut polytope of  $G_{\text{lin}}$ .

If  $P$  is the unit hypercube in (QP), solving the resulting QUBO problem thus amounts to determine a maximum cut in  $G_{\text{lin}}$ , i.e., to optimize over a cut polytope defined in the  $|E_{\text{lin}}|$ -dimensional space. If  $P$  is a strict subset of the unit hypercube, i.e., if additional constraints are present, these constraints can be transformed as well and we derive that  $P$  is isomorphic to a cut polytope with further linear constraints. In particular, all inequalities valid for the cut polytope still yield valid inequalities for the linearized binary quadratic problem.

Clearly, intersecting the cut polytope with arbitrary hyperplanes yields in general a non-integer polytope. The structure of the convex hull of integer points in the resulting polytope can be very different from a cut polytope. In this case it is not clear whether the knowledge about the cut polytope can help solving the linearized constrained optimization problem. Some applications exist in which no speed-up is observed when designing a solution algorithm based on the cut polytope if the inequalities cut through the interior of the polytope. However, for the applications studied here and for several further relevant applications, the linear constraints cut out a *face* of the cut polytope. This is an important observation as then the convex hull of the feasible solutions inherits the cut-polytope structure. Therefore, approaches exploiting the cut-polytope structure can lead to effective solution algorithms. This connection has been exploited in the work by Buchheim et al. [15] where several different applications were studied for which the linear constraints always induce faces of some cut polytope. General separation methods for constrained quadratic problems of form (QP) were designed that can complement or replace detailed polyhedral studies of the polytope of the linearized problem and that can be used as a black box. One of these routines consists of exploiting the connection to the maximum-cut problem by separating inequalities known to be valid for the cut polytope. The methods were tested on the quadratic linear ordering problem and the linear arrangement problem, among others. We will introduce these applications in the next section. The main contribution in [15] was to show that these general approaches lead to a dramatic decrease of both the number of nodes in the enumeration tree and the running time when compared to an algorithm that only uses the standard separation routines for the well-studied polytope  $P$  from (QP).

In the next section, we introduce the quadratic linear ordering problem which induces a face of a cut polytope. Subsequently, we observe that SRFLP is a special quadratic linear ordering problem.

### 29.2.2 The Quadratic Linear Ordering Polytope as a Face of a Cut Polytope

Let  $\pi = (\pi_1, \dots, \pi_n)$  denote a permutation of  $[n] := \{1, 2, \dots, n\}$ . The corresponding linear ordering variables  $x = (x_{ij}) \in \mathbf{R}^{n(n-1)}$  for  $i \neq j$  are defined as

$$x_{ij} = \begin{cases} 1, & \text{if } \pi_i < \pi_j \\ 0, & \text{if } \pi_j < \pi_i. \end{cases} \quad (29.1)$$

In practice, we can exploit  $x_{ij} = 1 - x_{ji}$  and thus eliminate half of the variables and only keep those with  $i < j$ . Given costs  $c_{ij} \in \mathbf{R}$  for each pair of facilities  $i, j$ , the linear ordering problem asks for optimizing a linear objective over the convex hull of all valid vectors  $x$ . This convex hull is called the linear ordering polytope  $P_{LO}$  [30, 59]. This is an NP-hard problem. For a recent survey on the linear ordering problem, we refer to the work by Charon and Hudry [22].

Allowing for products of linear ordering variables in the objective function, the corresponding quadratic linear ordering problem (QLO) in its general form is

$$\begin{aligned} \min & \sum_{(i,j,k,l) \in I} c_{ijkl} x_{ij} x_{kl} \\ (QLO) \quad \text{s.t.} & \quad x \in P_{LO} \\ & x_{ij} \in \{0, 1\} \quad \forall (i, j) \in J. \end{aligned}$$

The index set  $I$  consists of all quadruples  $(i, j, k, l)$  such that  $x_{ij} x_{kl}$  occurs as a product in the objective function, while  $J$  is the set of all pairs  $(i, j)$  for which a linear ordering variable  $x_{ij}$  is needed. As  $x_{ij}^2 = x_{ij}$  for any binary variable, (QLO) also takes linear terms into account.

In order to linearize the objective function, we introduce a new binary variable  $y_{ijkl}$  for each  $(i, j, k, l) \in I$ , modeling the product  $x_{ij} x_{kl}$ . (Note that  $y_{ijkl} = y_{klij}$ .) Applying the standard linearization, the corresponding linearized quadratic linear ordering problem (LQLO) can be written as

$$\begin{aligned} \min & \sum_{(i,j,k,l) \in I} c_{ijkl} y_{ijkl} \\ (LQLO) \quad \text{s.t.} & \quad x \in P_{LO} \\ & x_{ij} \in \{0, 1\} \quad \forall (i, j) \in J \\ & y_{ijkl} \leq x_{ij}, x_{kl} \quad \forall (i, j, k, l) \in I \\ & y_{ijkl} \geq x_{ij} + x_{kl} - 1 \quad \forall (i, j, k, l) \in I \\ & y_{ijkl} \in \{0, 1\} \quad \forall (i, j, k, l) \in I. \end{aligned}$$

Buchheim et al. [16] introduced the above model for the so-called bipartite crossing minimization problem. Additionally, a quadratic reformulation of the constraints defining  $P_{LO}$  was given: it was shown that a 0/1 vector  $(x, y)$  satisfying  $y_{ijkl} = x_{ij} x_{kl}$  is feasible for (LQLO) if and only if

$$x_{ik} - y_{ijik} - y_{ikjk} + y_{ijjk} = 0 \quad \forall i < j < k \text{ s.t. } (i, j, k, l) \in I \text{ for some } l. \quad (29.2)$$

Furthermore, the constraints (29.2) yield a minimum equation system for (LQLO). Equations (29.2) were used by Lewis et al. [45] in order to derive penalty functions for the original linear ordering problem. Note that (LQLO) is a quadratic binary optimization problem where the feasible solutions need to satisfy further side constraints, namely those restricting the set of feasible solutions to linear orderings. By proving validity of the corresponding inequality  $x_{ik} - y_{ijk} - y_{ikj} + y_{ijk} \geq 0$  for each of the equalities appearing in (29.2), it was shown in [16] that the hyperplanes (29.2) cut out faces of the cut polytope associated to graph  $G_{\text{lin}}$ . Exploiting this result, both IP- and SDP-based methods originally designed for maximum-cut problems were used in [16] to solve the quadratic linear ordering problem. It turned out that the SDP-based approach outperformed the IP-based techniques. In the recent work by Hungerländer and Rendl [39], (29.2) were also used for the quadratic linear ordering problem. The SDP-based method from [16] is improved by additionally separating matrixcuts that were introduced by Lovász and Schrijver in [48]. The method is used for different applications of quadratic linear ordering, the SRFLP being one of them. It turns out that the method can effectively solve very large instances to optimality.

### 29.2.3 The Connection of SRFLP and Related Problems to the Quadratic Linear Ordering Problem

In the following, we present an integer LP formulation of SRFLP. It will turn out that SRFLP is a special instance of a quadratic linear ordering problem.

Let  $\pi = (\pi_1, \dots, \pi_n)$  denote a permutation of the indices  $[n]$  of the facilities, so that the leftmost facility is  $\pi_1$ , the facility to the right of it is  $\pi_2$ , and so on, with  $\pi_n$  being the last facility in the arrangement. Given a permutation  $\pi$  and two distinct facilities  $i$  and  $j$ , the center-to-center distance between  $i$  and  $j$  with respect to this permutation is  $\frac{1}{2}\ell_i + D_\pi(i, j) + \frac{1}{2}\ell_j$ , where  $D_\pi(i, j)$  denotes the sum of the lengths of the facilities between  $i$  and  $j$  in the ordering defined by  $\pi$ . Solving the SRFLP consists of finding a permutation of the facilities that minimizes the weighted sum of the distances between all pairs of facilities. In other words, the problem is:

$$\min_{\pi \in \Pi_n} \sum_{i < j} c_{ij} \left[ \frac{1}{2}\ell_i + D_\pi(i, j) + \frac{1}{2}\ell_j \right],$$

where  $\Pi_n$  denotes the set of all permutations of  $[n]$ . Note that here and in all subsequent formulations of the SRFLP, the assumption that  $c_{ij} \geq 0$  ensures that the facilities are placed next to each other, i.e., without holes in the arrangement.

Simmons [65] observed that the crux of the problem is to minimize  $\sum_{i < j} c_{ij} D_\pi(i, j)$  over all permutations  $\pi \in \Pi_n$ . It is also clear that  $D_\pi(i, j) = D_{\pi'}(i, j)$ , where  $\pi'$  denotes the permutation symmetric to  $\pi$ , defined by  $\pi'_i = \pi_{n+1-i}$ ,  $i = 1, \dots, n$ . Hence,

it is possible to simplify the problem by considering only the permutations for which, say, facility 1 is on the left half of the arrangement. This type of symmetry-breaking strategy is important for reducing the computational requirements of most algorithms, including those based on LP or dynamic programming.

The SRFLP can be modeled as the optimization of a linear function over the distance polytope. This polytope is the convex hull of distance vectors  $d \in \mathbf{R}_+^{n \choose 2}$  that can be realized by some arrangement of objects. In [6] it is shown that the polytope lies in the cut cone, i.e., the convex cone generated by all incidence vectors of cuts. It was shown that several well-known facet-inducing inequalities for the latter also induce facets for the distance polytope. However, when solving the SRFLP this way, running times can grow quickly.

Alternatively, the sum of the lengths of the facilities between  $i$  and  $j$  in the ordering defined by  $\pi$  can be expressed in terms of products of linear ordering variables as follows. A facility  $k$  lies between  $i$  and  $j$  if and only if either  $i$  is before  $k$  and  $k$  is before  $j$ , or vice versa  $j$  is before  $k$  and  $k$  is before  $i$ . Thus,  $D(i, j)$  can be written as  $D(i, j) = \sum_k \ell_k x_{ik} x_{kj} + \sum_k \ell_k x_{jk} x_{ki}$ , where  $x_{ik}$  is the usual linear ordering variable modeling whether  $\pi_i < \pi_k$  or not. The objective function terms are then summed up appropriately, the resulting cost vector being called  $c'$ . Therefore, up to a constant, the SRFLP can be rewritten as a specific quadratic linear ordering problem of the form

$$\begin{aligned} \min \quad & \sum_{i \neq j \neq k \neq i} c'_{ij} x_{ik} x_{kj} \\ (\text{QLO}_2) \quad \text{s.t.} \quad & x \in P_{LO} \\ & x_{ij} \in \{0, 1\} \quad \forall i, j \in \{1 \dots n\}, i \neq j. \end{aligned}$$

If all object lengths and all non-zero weights equal 1, the application  $(\text{QLO}_2)$  is called linear arrangement problem. We noted above that the quadratic linear ordering problem is isomorphic to a face of a cut polytope. As a consequence, the same is true for the SRFLP and the linear arrangement problem.

We note that SRFLP and linear arrangement in the formulation  $(\text{QLO}_2)$  only require products of linear ordering variables of the form  $x_{ik} x_{kj}$ , whose number is only  $O(n^3)$ . All other products  $x_{ik} x_{lj}$  with pairwise different  $i, j, k, l$  have cost coefficients equal to zero.

A more concise model of the SRFLP is achieved when rewriting  $D(i, j)$  via betweenness variables. In fact, for the SRFLP it is not necessary to model for each pair of objects their relation in the ordering. It is enough to know for each pair of objects which facilities are *between* them. Betweenness variables  $\zeta_{ijk}$  for facilities  $i, j, k$  are defined as

$$\zeta_{ijk} = \begin{cases} 1, & \text{if department } k \text{ lies between departments } i \text{ and } j, \\ 0, & \text{otherwise.} \end{cases}$$

The betweenness polytope is the convex hull of the incidence vectors of feasible solutions, i.e.,

$$P_{\text{Bet}} = \text{conv} \left\{ \zeta \in \{0, 1\}^{n(n-1)(n-2)} : \zeta \text{ arises from a permutation of } 1, \dots, n \right\}.$$

Thus, by rewriting  $D(i, j) = \sum_k \ell_k \zeta_{ijk}$ , the SRFLP is the problem of optimizing a linear function over the betweenness polytope. The latter is well understood, see e.g. Christof et al. [23]. A betweenness variable  $\zeta_{ijk}$  can be written via linear ordering variables as  $\zeta_{ijk} = x_{ik}x_{kj} + x_{jk}x_{ki}$ . Therefore, the betweenness problem is a quadratic linear ordering problem in which all products  $x_{ij}x_{kl}$  with pairwise different  $i, j, k, l$  necessarily have zero contribution to the objective function. As the betweenness problem is a specific quadratic linear ordering problem in which only  $O(n^3)$  cost coefficients can take non-zero cost value, we immediately obtain that the betweenness polytope induces a face of a cut polytope.

Recently, a different relation of the betweenness polytope to the cut polytope was pointed out by Caprara et al. [17]. To this end, an undirected graph  $G_{\text{Bet}} = (V, E)$  is set up that contains a node for each facility. An edge  $(i, j) \in E$  between two facilities exists in  $G_{\text{Bet}}$  if the cost  $c_{ij}$  associated with  $i$  and  $j$  is nonzero. The authors observe that for fixed  $k$ , a betweenness incidence vector  $(\zeta_{ijk})$  is the incidence vector of a cut in the graph  $G_{\text{Bet}} \setminus \{k\}$ . In the latter, node  $k$  together with all its incident edges is deleted from the graph  $G_{\text{Bet}}$ . As a consequence, the projection of the betweenness variables  $\zeta_{ijk}$  for fixed  $k$  is isomorphic to the cut polytope associated with graph  $G_{\text{Bet}} \setminus \{k\}$ . Based on this observation, Caprara et al. [17] suggest an exact solution approach for the linear arrangement problem. One of their separation routines consists of separating odd-cycle inequalities in  $G_{\text{Bet}} \setminus \{k\}$ , for each  $k$ . Odd-cycle inequalities are known to be facet-defining for the cut polytope. A similar exact branch-and-cut approach could be designed for the slightly more general SRFLP.

In summary, there exist different connections of SRFLP to the cut polytope and the cut cone. The model by Letchford et al. [6] relates SRFLP to the cut cone by observing that the latter contains the distance polytope. If the SRFLP is modeled via the quadratic linear ordering problem, a face of a cut polytope is induced. The latter polytope is defined for the graph  $G_{\text{lin}}$ . Using the betweenness formulation instead, for each object  $k$  the projection of the problem is isomorphic to the cut polytope on the graph  $G_{\text{Bet}} \setminus \{k\}$ .

We note that the model via quadratic linear ordering is more general than the betweenness approach in the sense that every betweenness variable can be expressed via products of linear ordering variables, but not vice versa.

In the next section, we present the currently most effective LP-based approach by Amaral [4] which is based on a betweenness model. In Sect. 29.2.5, we obtain matrix-based formulations and SDP relaxations using the quadratic linear ordering model (QLO<sub>2</sub>).

### 29.2.4 The Betweenness-Based LP Formulation of Amaral for SRFLP

Although not explicitly stated, Amaral's model is implicitly based on the connection to the cut polytope as observed by Caprara et al. [17]. We explain this in more detail in the following. Amaral [4] used the betweenness formulation based on variables  $\zeta_{ijk}$  for SRFLP. The objective function of the SRFLP can then be expressed as

$$\sum_{i < j} c_{ij} \left[ \frac{1}{2} (\ell_i + \ell_j) + \sum_{k \neq i, j} \ell_k \zeta_{ijk} \right] \quad (29.3)$$

Let an index of an object be denoted by  $i, j, k$ , or  $d$ . To obtain an LP relaxation of the SRFLP, Amaral proposes to optimize (29.3) subject to the following partial description of  $P_{\text{Bet}}$  in which the objects  $i, j, k, d$  are always chosen pairwise different:

$$\zeta_{ijk} + \zeta_{ikj} + \zeta_{jki} = 1, \quad \text{for all } \{i, j, k\} \subseteq \{1, \dots, n\}, \quad (29.4)$$

$$\zeta_{ijd} + \zeta_{jkd} - \zeta_{ikd} \geq 0, \quad \text{for all } \{i, j, k, d\} \subseteq \{1, \dots, n\}, \quad (29.5)$$

$$\zeta_{ijd} + \zeta_{jkd} + \zeta_{ikd} \leq 2, \quad \text{for all } \{i, j, k, d\} \subseteq \{1, \dots, n\}, \quad (29.6)$$

$$0 \leq \zeta_{ijk} \leq 1, \quad \text{for all } \{i, j, k\} \subseteq \{1, \dots, n\}. \quad (29.7)$$

Equation (29.4) can easily be understood from the fact that the equalities  $\zeta_{ijk} = 1$ ,  $\zeta_{ikj} = 1$ , and  $\zeta_{jki} = 1$  are mutually exclusive (by definition of  $\zeta_{ijk}$ ). Using the equivalence  $\zeta_{ijk} \equiv x_{ik}x_{kj} + x_{jk}x_{ki}$  based on linear ordering variables, it follows that (29.4) is equivalent to the minimum equation system (29.2) known for the quadratic linear ordering problem. Without loss of generality, we assume  $i < j < k$  and rewrite (29.4) as

$$\begin{aligned} \zeta_{ijk} + \zeta_{ikj} + \zeta_{jki} = 1 &\Leftrightarrow (x_{ik}x_{kj} + x_{jk}x_{ki}) + (x_{ij}x_{jk} + x_{kj}x_{ji}) \\ &\quad + (x_{ji}x_{ik} + x_{ki}x_{ij}) = 1 \\ &\Leftrightarrow x_{ik}(1 - x_{jk}) + x_{jk}(1 - x_{ik}) + x_{ij}x_{jk} + \\ &\quad (1 - x_{jk})(1 - x_{ij}) + (1 - x_{ij})x_{ik} + (1 - x_{ik})x_{ij} = 1 \\ &\Leftrightarrow 2(x_{ik} - x_{ij}x_{ik} - x_{ik}x_{jk} + x_{ij}x_{jk}) = 0. \end{aligned}$$

Using the recent observation of Caprara et al. [17] that for a fixed object  $d$ , the incidence vectors of betweenness vectors are incidence vectors of cuts in the graph  $G_{\text{Bet}} \setminus \{d\}$ , it can be checked that (29.5) and (29.6) are precisely the triangle inequalities known to be facet-defining for the cut polytope. Indeed, projecting the inequalities accordingly and denoting the variables by  $\zeta^p$ , (29.5) and (29.6) read for pairwise different  $i, j, k$ :

$$\zeta_{ij}^p + \zeta_{jk}^p - \zeta_{ik}^p \geq 0, \quad \text{for all } \{i, j, k\} \subseteq \{1, \dots, n\}, \quad (29.8)$$

$$\zeta_{ij}^p + \zeta_{jk}^p + \zeta_{ik}^p \leq 2, \quad \text{for all } \{i, j, k\} \subseteq \{1, \dots, n\}, \quad (29.9)$$

These are exactly the triangle inequalities stating that a cut either contains zero or two edges from a triangle. They can be generalized to the odd-cycle inequalities that state in algebraic terms that a cycle and a cut can only coincide in an even number of common edges, see e.g. [26]. Alternatively, one can also check that (29.5) and (29.6) are the triangle inequalities by using the equivalence  $\zeta_{ijk} \equiv x_{ik}x_{kj} + x_{jk}x_{ki}$  and working through the classical linear transformation between the maximum-cut problem and QUBO.

Amaral proposes a further class of valid inequalities for  $P_{\text{Bet}}$  that can be used as cuts to improve the LP relaxation. They are as follows:

**Proposition 29.1.** [4] Let  $\beta \leq n$  be a positive even integer and let  $S \subseteq \{1, \dots, n\}$  such that  $|S| = \beta$ . For each  $r \in S$ , and for any partition  $(S_1, S_2)$  of  $S \setminus \{r\}$  such that  $|S_1| = \frac{1}{2}\beta$ , the inequality

$$\sum_{t < q, t \in S_1, q \in S_1} \zeta_{tqr} + \sum_{t < q, t \in S_2, q \in S_2} \zeta_{tqr} - \sum_{t \in S_1, q \in S_2} \zeta_{\min\{t,q\}, \max\{t,q\}, r} \leq 0 \quad (29.10)$$

is valid for  $P_{\text{Bet}}$ .

It is easy to check that for  $\beta = 4$ , (29.10) is a triangle inequality of the form (29.5).

Again, the inequalities (29.10) can be interpreted as inequalities for the cut polytope. For the corresponding graph  $G_{\text{Bet}} \setminus \{r\}$ , the inequality (29.10) takes the form

$$\zeta^p(E(S_1)) + \zeta^p(E(S_2)) - \zeta^p(\delta(S_1)) \leq 0 \quad (29.11)$$

where for a node set  $U$  the usual abbreviation  $\zeta^p(E(U)) = \sum_{e \in E(U)} \zeta_e^p$  is used. The inequality (29.11) is a so-called clique inequality [26] that is here switched along the cut  $\delta(S_1)$ . Clique inequalities are known to be valid and facet-inducing for the cut polytope. Their exact separation is however NP-hard, and so often heuristic separation procedures are used in practice. Amaral does not use a general separation routine for arbitrary values of  $\beta$  in (29.10) but proposes separating inequalities with  $\beta = 6$  by enumeration. The corresponding class of clique inequalities are often called pentagonal inequalities.

Unlike for the LP relaxation, the triangle inequalities are not included explicitly in the SDP relaxations that are presented in Sect. 29.2.5. However, they can be added as valid inequalities there, and so can any other valid inequalities for the cut polytope. Furthermore, theoretical results in Sect. 29.2.7 below show that when certain triangle inequalities are present in either the LP or the SDP relaxation, a number of the pentagonal and of the so-called hexagonal inequalities automatically hold as well (see Sect. 29.2.7 for the definition of these classes of inequalities).

Amaral also shows that the size of the LP relaxation can be reduced by projecting the feasible set into a lower-dimensional space. We refer the reader to [4] for details. Very recently, Sanjeevi and Kianfar [63] have independently shown that several

of the valid inequalities from Amaral are indeed facet-defining. The classes of inequalities studied there are well known to be facet-defining for the cut polytope. It is an interesting fact that they remain facet-inducing when considering only the convex hull of the SRFLP incidence vectors which induce a face of this cut polytope.

### 29.2.5 Matrix-Based Formulations and SDP Relaxations

In the sections above, we discussed integer LP-based solution approaches where it is convenient to formulate models in binary variables that either take value zero or value one. In this section, we introduce SDP relaxations where the variables are usually assumed to take values in  $\{\pm 1\}$ . We start by presenting the quadratic formulation of the SRFLP proposed by Anjos et al. [7]. The latter is basically a modeling via the quadratic linear ordering problem given in (QLO<sub>2</sub>) but written in  $\pm 1$  instead of binary variables. Their model can be described as follows.

For a given permutation  $\pi$  of  $[n]$ , for each pair of integers  $ij$  with  $1 \leq i < j \leq n$ , define a binary  $\pm 1$  variable such that

$$R_{ij} := \begin{cases} 1, & \text{if } \pi_j < \pi_i, \\ -1, & \text{if } \pi_i < \pi_j. \end{cases}$$

In this definition, the order of the subscripts matters, and  $R_{ij} = -R_{ji}$ . In fact,  $R_{ij}$  determines a linear ordering on the objects.

Given a particular assignment of  $\pm 1$  values to the  $R_{ij}$  variables, if this assignment represents a permutation of  $[n]$ , then the condition that

if  $i$  is to the right of  $j$  and  $j$  is to the right of  $k$ , then  $i$  is to the right of  $k$

must be enforced. Equivalently, if  $R_{ij} = R_{jk}$  then  $R_{ik} = R_{ij}$ . This necessary transitivity condition can be formulated as a set of quadratic constraints:

$$R_{ij}R_{jk} - R_{ij}R_{ik} - R_{ik}R_{jk} = -1 \text{ for all triples } 1 \leq i < j < k \leq n. \quad (29.12)$$

Interestingly, (29.12) is exactly the set of equations (29.2), written in  $\pm 1$  variables. After (29.12) was introduced in [7] for SRFLP, it was shown in [16] that the latter is a minimum equation system for the quadratic linear ordering problem (QLO).

The objective function of the SRFLP can be expressed as

$$\sum_{i < j} c_{ij} \left[ \frac{1}{2} (\ell_i + \ell_j) + \sum_{k \neq i, j} \ell_k \left( \frac{1 - R_{ki}R_{kj}}{2} \right) \right]$$

upon observing that  $R_{ki}R_{kj} = -1$  if and only if facility  $k$  is between  $i$  and  $j$ . The resulting formulation of the SRFLP is:

$$\begin{aligned} \min \quad & K - \sum_{i < j} \frac{c_{ij}}{2} \left[ \sum_{k < i} \ell_k R_{ki} R_{kj} - \sum_{i < k < j} \ell_k R_{ik} R_{kj} + \sum_{k > j} \ell_k R_{ik} R_{jk} \right] \\ \text{s.t.} \quad & R_{ij} R_{jk} - R_{ij} R_{ik} - R_{ik} R_{jk} = -1 \text{ for all triples } i < j < k \\ & R_{ij}^2 = 1 \text{ for all } i < j \end{aligned} \quad (29.13)$$

where  $K := \left( \sum_{i < j} \frac{c_{ij}}{2} \right) \left( \sum_{k=1}^n \ell_k \right)$ . Note that if every  $R_{ij}$  variable is replaced by its negative, then there is no change whatsoever to the formulation. For this reason, the formulation (29.13) and the subsequent matrix-based formulations and corresponding SDP relaxations implicitly account for the symmetry in the SRFLP.

Let  $\rho \in \{\pm 1\}^{\binom{n}{2}}$  denote an assignment of values to the  $R_{ij}$  variables. Hence define the set:

$$\mathcal{R}_n := \left\{ \rho \in \{\pm 1\}^{\binom{n}{2}} : R_{ij} R_{jk} - R_{ij} R_{ik} - R_{ik} R_{jk} = -1 \quad \forall 1 \leq i < j < k \leq n \right\}$$

and consider the function  $f : \mathcal{R}_n \rightarrow \Pi_n$  defined by

$$f(\rho) = (\pi_1, \dots, \pi_n), \text{ where } \pi_k := \frac{P_k + n + 1}{2}$$

and

$$P_k := \sum_{j \neq k} R_{kj} = \sum_{j < k} -R_{jk} + \sum_{j > k} R_{kj} \quad \text{for } k = 1, 2, \dots, n. \quad (29.14)$$

We have the following fact:

**Theorem 29.1.** *The function  $f : \mathcal{R}_n \rightarrow \Pi_n$  is a bijection.*

This result follows from Buchheim et al. [16] and the earlier work of Murata et al. [56] who demonstrated that for every layout in two dimensions, there exists a finite representation via so-called sequence-pairs. This technique has been applied with great success in VLSI floorplanning, the original application in [56], and also in two-dimensional facility layout [52, 71]. An independent proof in terms of the variables and structure used here was given in Anjos and Yen [11].

Following [7], we obtain a formulation of the SRFLP in the space of real symmetric matrices by fixing an ordering of all pairs  $ij$  such that  $i < j$ , and defining the vector

$$v := (R_{p_1}, \dots, R_{p_{\binom{n}{2}}})^T,$$

where  $p_k$  denotes the  $k^{\text{th}}$  pair  $ij$  in the ordering. Using  $v$ , we construct the rank-one matrix  $X := vv^T$  whose rows and columns are indexed by pairs  $ij$ . By construction,

$X_{p_i, p_j} = R_{p_i} R_{p_j}$  for any two pairs  $p_i, p_j$ , and therefore we can formulate the SRFLP as:

$$\begin{aligned} \min \quad & K - \sum_{i < j} \frac{c_{ij}}{2} \left[ \sum_{k < i} \ell_k X_{ki,kj} - \sum_{i < k < j} \ell_k X_{ik,kj} + \sum_{k > j} \ell_k X_{ik,jk} \right] \\ \text{s.t.} \quad & X_{ij,jk} - X_{ij,ik} - X_{ik,jk} = -1 \text{ for all triples } i < j < k \\ & \text{diag}(X) = e \\ & \text{rank}(X) = 1 \\ & X \geq 0 \end{aligned} \quad (29.15)$$

where  $\text{diag}(X)$  represents a vector containing the diagonal elements of  $X$ ,  $e$  denotes the vector of all ones, and  $X \geq 0$  denotes that  $X$  is symmetric positive semidefinite. Note that  $X \in \mathcal{S}^{\binom{n}{2}}$ , the set of symmetric matrices of dimension  $\binom{n}{2}$ .

Removing the rank constraint from (29.15) yields an SDP relaxation. Note that in general the SDP problem only provides a lower bound on the optimal value of the SRFLP, and not a feasible solution, unless the optimal matrix  $X^*$  happens to have rank equal to one. A standard way to tighten linear or semidefinite relaxations of integer optimization problems is, as mentioned earlier, to add inequalities (such as the triangle inequalities) that are valid for the integer feasible points. The SDP relaxation of (29.15) with a straightforward scheme to iteratively add violated triangle inequalities was used in [10] to solve SRFLPs with up to 30 facilities to global optimality for the first time.

From a computational perspective, the main limitation of the SDP relaxation of (29.15) is that it has  $O(n^3)$  linear constraints; this limits the size of instances that can be tackled with it. This motivated Anjos and Yen [11] to consider an alternative formulation, obtained by reducing the number of the linear constraints in the following way:

$$\begin{aligned} \min \quad & K - \sum_{i < j} \frac{c_{ij}}{2} \left[ \sum_{k < i} \ell_k X_{ki,kj} - \sum_{i < k < j} \ell_k X_{ik,kj} + \sum_{k > j} \ell_k X_{ik,jk} \right] \\ \text{s.t.} \quad & \sum_{k=1, k \neq i, j}^n X_{ij,jk} - \sum_{k=1, k \neq i, j}^n X_{ij,ik} - \sum_{k=1, k \neq i, j}^n X_{ik,jk} = -(n-2) \text{ for all } i < j \\ & \text{diag}(X) = e \\ & \text{rank}(X) = 1 \\ & X \geq 0. \end{aligned} \quad (29.16)$$

It is straightforward to prove that the two formulations are equivalent (but not their SDP relaxations).

**Theorem 29.2.** [11] The feasible sets of (29.15) and (29.16) are equal.

Removing the rank-one constraint from (29.16) again yields an SDP relaxation. Although the number of linear constraints in this relaxation is now  $O(n^2)$ , with a corresponding favourable impact on the computational time and memory requirements

**Table 29.1** Bounds for some extremely large instances

Number of departments	Number of instances	Range of gaps	Average CPU time
56	5	3.49–4.47%	3 h 03 m
64	5	2.80–5.15%	9 h 00 m
72	5	2.89–4.03%	23 h 20 m
81	5	3.44–4.97%	2 d 0 h 24 m
100	5	3.18–3.81%	9 d 17 h 0 m

of a primal-dual interior-point algorithm, it turns out that the quality of the solution appears to deteriorate only slightly, yielding optimality gaps not greater than 5% for instances with up to  $n = 100$ . A summary of the results reported in [11] is provided in Table 29.1.

### 29.2.6 SDP-Based Heuristic to Obtain a Layout

The SDP relaxations of (29.15) and (29.16) are closely related to the basic SDP relaxation for the maximum-cut problem used in the ground-breaking paper of Goemans and Williamson [29]. However, we cannot use their randomized hyperplane rounding procedure because it does not guarantee that we will obtain a valid representation of a permutation. This is because it does not ensure that the equality constraints (29.12) hold, and hence the result may not represent a valid permutation. One possibility is to use a heuristic to fix the result, as is done in [39]. Alternatively, a different rounding procedure, based on Theorem 29.1, was proposed in [7] to obtain a permutation from the optimal solution to the SDP relaxation.

The procedure is as follows: If  $X^*$  is the optimal solution to the SDP relaxation, then each row of  $X^*$  corresponds to a specific pair  $i_1 j_1$  of facilities. Therefore, for any row of  $X^*$ , if we set  $R_{i_1 j_1} = +1$ , then we can scan the other entries of the row and assign the value  $X_{i_1 j_1, i_2 j_2}$  to the variable  $R_{i_2 j_2}$ , for every pair  $i_2 j_2 \neq i_1 j_1$ . (Note that every value assigned to an  $R_{ij}$  is in the interval  $[-1, 1]$  because every feasible  $X$  for the SDP is a correlation matrix.) Using these values, we compute

$$\omega_k = \frac{1}{2} \left( n + 1 + \sum_{j \neq k} R_{kj} \right)$$

for  $k = 1, \dots, n$ .

The motivation for the values  $\omega_k$  comes from the fact that if  $X^*$  is rank-one, then  $\omega_k = \pi_k, k = 1, \dots, n$ , where the  $\pi_k$  define the bijection in Theorem 29.1, and hence the  $\omega_k$  define a permutation of  $[n]$ . In general,  $\text{rank}(X^*) > 1$  and thus  $\omega_k \in [1, n]$ , so the SDP-based heuristic obtains a permutation of  $[n]$  by sorting the values  $\omega_k$ . The sorting can be in either decreasing or increasing order (since the objective value is

the same), and since the procedure implicitly sets  $R_{i_1 j_1} = +1$ , we choose the order that places  $i_1$  to the right of  $j_1$ . The output of the heuristic is the best layout found by considering every row in turn.

### 29.2.7 On the Facial Structure of the Relaxations

Consider the following set of solutions feasible for both the SDP relaxation of (29.15) and the LP relaxation (29.4)–(29.7) under the aforementioned mapping between their variables:

$$\mathcal{X}_n := \left\{ X \in \mathcal{S}^{\binom{n}{2}} : -1 \leq X_{ij,kl} \leq 1, \right. \\ \left. X_{ij,jk} - X_{ij,ik} - X_{ik,jk} = -1 \quad \forall 1 \leq i < j < k \leq n \right\}.$$

It is straightforward to check that a number of triangle inequalities automatically hold for every  $X \in \mathcal{X}_n$ .

**Lemma 29.1.** *If  $X \in \mathcal{X}_n$ , then for every triple of pairs  $(i_1, i_2)$ ,  $(i_1, i_3)$ , and  $(i_2, i_3)$ , where  $i_1 < i_2 < i_3$ , the entries of  $X$  satisfy*

$$X_{i_1 i_2, i_1 i_3} + X_{i_1 i_2, i_2 i_3} + X_{i_1 i_3, i_2 i_3} \geq -1, \\ X_{i_1 i_2, i_1 i_3} - X_{i_1 i_2, i_2 i_3} - X_{i_1 i_3, i_2 i_3} \geq -1, \\ -X_{i_1 i_2, i_1 i_3} - X_{i_1 i_2, i_2 i_3} + X_{i_1 i_3, i_2 i_3} \geq -1, \\ -X_{i_1 i_2, i_1 i_3} + X_{i_1 i_2, i_2 i_3} - X_{i_1 i_3, i_2 i_3} \geq -1.$$

*Proof.* Since  $X \in \mathcal{X}_n$ , we know  $X_{i_1 i_2, i_2 i_3} - X_{i_1 i_2, i_1 i_3} - X_{i_1 i_3, i_2 i_3} = -1$ , and therefore the fourth inequality trivially holds. Now,

$$X_{i_1 i_2, i_2 i_3} + X_{i_1 i_2, i_1 i_3} + X_{i_1 i_3, i_2 i_3} = X_{i_1 i_2, i_2 i_3} + X_{i_1 i_2, i_1 i_3} + (1 + X_{i_1 i_2, i_2 i_3} - X_{i_1 i_2, i_1 i_3}) \\ = 1 + 2X_{i_1 i_2, i_2 i_3} \geq -1,$$

and hence the first triangle inequality above holds. The other two inequalities follow similarly.  $\square$

We have thus observed that  $4\binom{n}{3}$  triangle inequalities of the form (29.8–29.9) automatically hold for all the feasible matrices of  $\mathcal{X}_n$ . If we consider the addition of the remaining triangle inequalities, we obtain the following (tighter) set of feasible solutions:

$$\mathcal{X}_n^\Delta = \mathcal{X}_n \cap \left\{ X \in \mathcal{S}^{\binom{n}{2}} : X_{p_1, p_2} + X_{p_1, p_3} + X_{p_2, p_3} \geq -1, X_{p_1, p_2} \right. \\ \left. - X_{p_1, p_3} - X_{p_2, p_3} \geq -1, \right. \\ \left. -X_{p_1, p_2} - X_{p_1, p_3} + X_{p_2, p_3} \geq -1, -X_{p_1, p_2} + X_{p_1, p_3} - X_{p_2, p_3} \geq -1, \right. \\ \left. \forall p_1, p_2, p_3 : \{p_1, p_2, p_3\} \neq \{(i_1, i_2), (i_1, i_3), (i_2, i_3)\} \right. \\ \left. \text{for any } i_1 < i_2 < i_3 \right\}.$$

We now show that a number of facet-defining inequalities of the cut polytope are implicitly enforced in  $\mathcal{X}_n^\Delta$ .

First, we consider the pentagonal inequalities. For each subset of pairs  $\{p_1, \dots, p_5\}$  corresponding to rows and columns of  $X$ , there are 16 such inequalities and they can be represented as:

$$\sum_{1 \leq i < j \leq 5} \delta_i \delta_j X_{p_i, p_j} \geq -2,$$

where  $\delta_k \in \{-1, 1\}, k = 1, 2, 3, 4, 5$ . Hence there are  $16 \binom{\binom{n}{2}}{5}$  valid pentagonal inequalities in total. It is proved in [8] that for  $\mathcal{X}_n^\Delta$ ,  $90 \binom{n}{4}$  of those inequalities automatically hold.

**Lemma 29.2.** [8] Suppose that  $X \in \mathcal{X}_n^\Delta$ , and (from the definition of  $\mathcal{X}_n^\Delta$ ) consider any five pairs  $p_1, \dots, p_5$  that satisfy

$$X_{p_1, p_4} - X_{p_1, p_2} - X_{p_2, p_4} = -1, \quad (29.17)$$

$$X_{p_1, p_5} - X_{p_1, p_3} - X_{p_3, p_5} = -1. \quad (29.18)$$

Then for all choices of  $\delta_i \in \{-1, 1\}, i = 1, 2, 3, 4, 5$ , such that  $(\delta_1 \delta_2 - 1)(\delta_1 \delta_3 - 1)(\delta_1 \delta_4 + 1)(\delta_1 \delta_5 + 1) = 0$ , the pentagonal inequality

$$\sum_{1 \leq i < j \leq 5} \delta_i \delta_j X_{p_i, p_j} \geq -2$$

holds. This gives a total of 15 pentagonal inequalities.

Lemma 29.2 leads to the following theorem.

**Theorem 29.3.** [8] Suppose that  $X \in \mathcal{X}_n^\Delta$ , and consider any choice  $1 \leq i_1 < i_2 < i_3 < i_4 \leq n$ . Then for each of the following sets of row indices for  $X$ , precisely 15 pentagonal inequalities hold:

$$\begin{aligned} &\{(i_1, i_2), (i_1, i_3), (i_1, i_4), (i_2, i_3), (i_2, i_4)\}, \\ &\{(i_1, i_2), (i_1, i_3), (i_1, i_4), (i_2, i_3), (i_3, i_4)\}, \\ &\{(i_1, i_2), (i_1, i_3), (i_1, i_4), (i_2, i_4), (i_3, i_4)\}, \\ &\{(i_1, i_2), (i_1, i_3), (i_2, i_3), (i_2, i_4), (i_3, i_4)\}, \\ &\{(i_1, i_2), (i_1, i_4), (i_2, i_3), (i_2, i_4), (i_3, i_4)\}, \text{ and} \\ &\{(i_1, i_3), (i_1, i_4), (i_2, i_3), (i_2, i_4), (i_3, i_4)\}. \end{aligned}$$

Hence, for  $X \in \mathcal{X}_n^\Delta$ , the  $90 \binom{n}{4}$  pentagonal inequalities described above automatically hold.

One can also consider the hexagonal inequalities: They are the inequalities obtained from

$$2 \sum_{t=2}^6 \delta_1 \delta_t X_{p_1, p_t} + \sum_{2 \leq s < t \leq 6} \delta_s \delta_t X_{p_s, p_t} \geq -4$$

by permutation of the pairs and where  $\delta_k \in \{-1, 1\}, k = 1, \dots, 6$ , for every 6-tuple of pairs  $p_1, \dots, p_6$ .

Using the same approach as for proving Theorem 29.3, one can show that a number of hexagonal inequalities are implicitly enforced.

**Theorem 29.4.** [8] Suppose that  $X \in \mathcal{X}_n^\Delta$ , and that for the set of pairs  $\{p_1, \dots, p_6\}$  corresponding to rows and columns of  $X$ , the following hold:

$$X_{p_1, p_4} - X_{p_1, p_2} - X_{p_2, p_4} = -1, \quad X_{p_1, p_5} - X_{p_1, p_3} - X_{p_3, p_5} = -1,$$

$$X_{p_2, p_6} - X_{p_2, p_3} - X_{p_3, p_6} = -1, \text{ and } X_{p_4, p_6} - X_{p_4, p_5} - X_{p_5, p_6} = -1.$$

Then for all choices of  $\delta_k \in \{-1, 1\}, k = 1, 2, 3, 4, 5, 6$ , the hexagonal inequalities

$$2 \sum_{j \neq i, j=1}^6 \delta_i \delta_j X_{p_i, p_j} + \sum_{j \neq i, k \neq i, 1 \leq j < k \leq 6} \delta_j \delta_k X_{p_j, p_k} \geq -4, \quad i = 1, \dots, 6,$$

hold.

Hence,  $192 \binom{n}{4}$  hexagonal inequalities automatically hold for  $X \in \mathcal{X}_n^\Delta$ .

### 29.3 VLSI Floorplanning

In this section, we introduce the application of SDP to the VLSI macrocell floorplanning problem. The general VLSI floorplanning problem consists of arranging a set of rectangular modules on a rectangular chip area so that an appropriate measure of performance is optimized. The resulting layout is called a floorplan. In the macrocell context, the modules are soft, meaning that the area of each rectangular module is assumed to be fixed while its height and width are allowed to vary subject to given aspect ratio constraints [62, 70]. The floorplanning of soft modules is an important problem because it takes advantage of the fact that at this stage of the physical design process, the rectangular modules have not themselves been laid out in detail yet, and so the floorplanner can do a better job if it is allowed to change the dimensions of the modules in a controlled manner.

Specifically, the VLSI macrocell floorplanning problem consists of partitioning a given rectangular chip into  $N$  modules with fixed areas so as to minimize the total cost associated with interactions between these modules and with  $N_p$  fixed I/O pads located all around the chip. For this problem, Takouda et al. [69] presented the first formulation using a mixed-integer conic optimization model motivated by the success of the SDP approach to SRFLP.

### 29.3.1 A Mixed-Integer Conic Optimization Formulation for Floorplanning

The VLSI macrocell surface is modeled as a  $w_F \times h_F$  rectangle with given area  $a_F$ . We set the origin of our system of coordinates at the center of the chip. A module  $i$  is a  $w_i \times h_i$  rectangle with given area  $a_i$  and centroid  $(x_i, y_i)$ . We denote by  $w_i^{\max}$  (and respectively  $w_i^{\min}, h_i^{\max}, h_i^{\min}$ ) the given maximum length (resp. minimum length, maximum width, minimum width) of each module. The pads are determined by their coordinates, and are denoted using the letters  $p, q, r$  while the modules are denoted by  $i, j, k$ . We finally assume that the cost of the connection per unit of distance between modules  $i, j$  is denoted by  $c_{ij}$  and the cost per unit of distance between a module  $i$  and a pad  $p$  is  $\gamma_{ip}$ . The variables are  $(x_i, y_i)$ ,  $h_i$  and  $w_i$  for each module  $i$ , and the objective is to minimize the total weighted connection cost.

#### 29.3.1.1 Area Constraints

We relax the area constraint  $w_i h_i = a_i$  for each module as

$$w_i h_i \geq a_i. \quad (29.19)$$

This convex relaxation can be expressed as a semidefinite constraint:

$$\begin{pmatrix} w_i & \sqrt{a_i} \\ \sqrt{a_i} & h_i \end{pmatrix} \succeq 0, \quad (29.20)$$

which is well defined since  $a_i > 0$ . Convexifying the area constraints is a reasonable approach for floorplanning in view of the following lemma, and the fact that the condition  $\sum_i a_i = a_F$  holds for nearly all macrocell floorplanning problems.

**Lemma 29.3.** [69] If (29.20) holds and  $a_F = \sum_i a_i$ , then the area constraints  $w_i h_i = a_i$  are satisfied for every  $i$ .

From a practical perspective, we point out that each  $2 \times 2$  semidefinite constraint (29.20) can be expressed as a second-order cone constraint of size 3 [42]. Using second-order cone constraints normally leads to lower running times in practice.

#### 29.3.1.2 Fit-in-the-Chip Constraints

For module  $i$  to lie inside the chip, the following inequalities have to be satisfied:

$$-\frac{1}{2}(w_F - w_i) \leq x_i \leq \frac{1}{2}(w_F - w_i), \quad (29.21)$$

$$-\frac{1}{2}(h_F - h_i) \leq y_i \leq \frac{1}{2}(h_F - h_i). \quad (29.22)$$

If desired, they can also be expressed as semidefinite constraints since (29.21) follows from:

$$x_i^2 \leq \left[ \frac{1}{2}(w_F - w_i) \right]^2 \Leftrightarrow \begin{pmatrix} \frac{1}{2}(w_F - w_i) & x_i \\ x_i & \frac{1}{2}(w_F - w_i) \end{pmatrix} \succeq 0, \quad (29.23)$$

provided  $0 \leq w_i \leq w_F$ . Similarly (29.22) follows from:

$$\begin{pmatrix} \frac{1}{2}(h_F - h_i) & y_i \\ y_i & \frac{1}{2}(h_F - h_i) \end{pmatrix} \succeq 0, \quad (29.24)$$

provided  $0 \leq h_i \leq h_F$ . Computationally, however, the linear constraints (29.21–29.22) lead to lower running times.

### 29.3.1.3 Aspect Ratio Constraints

The aspect ratio of department  $i$  is defined as  $\frac{\max\{h_i, w_i\}}{\min\{h_i, w_i\}}$ . In facility layout problems, it is often desirable to set bounds on the aspect ratios of the departments to ensure that no department is excessively narrow (in either direction) in the computed layout. However, as the bounds on the aspect ratios become smaller, the layout problem becomes more constrained and the total cost of the optimal solution increases. VLSI floorplanning benchmarks typically include aspect ratio restrictions on the modules.

Given aspect ratio bounds  $\beta_i \geq 1$  for each module  $i$ , the aspect ratio requirements are  $\frac{w_i}{h_i} \leq \beta_i$  and  $\frac{h_i}{w_i} \leq \beta_i$  and can be enforced using linear constraints  $w_i \leq \beta_i h_i$  and  $h_i \leq \beta_i w_i$  for fixed  $\beta_i$ . Alternatively, since  $w_i > 0, h_i > 0, a_i > 0$ , and  $w_i h_i = a_i$ , these constraints are equivalent to  $w_i^2 \leq \beta_i a_i$  and  $h_i^2 \leq \beta_i a_i$ , and therefore to

$$\begin{pmatrix} \beta_i & w_i \\ w_i & a_i \end{pmatrix} \succeq 0, \quad \begin{pmatrix} \beta_i & h_i \\ h_i & a_i \end{pmatrix} \succeq 0. \quad (29.25)$$

These constraints can also be expressed as second-order cone constraints with small support. The formulation (29.25) has the advantage that it allows the possibility of letting  $\beta_i$  be a decision variable, which can be useful in practice.

### 29.3.1.4 Non-overlap Constraints

Finally, we consider the non-overlap constraints. These disjunctive constraints require the modules to be separated in either the  $x$  or the  $y$  direction. They can be expressed as follows:

$$\underbrace{d_{ij}^x}_{=|x_i - x_j|} \geq \frac{1}{2}(w_i + w_j) \quad \text{or} \quad \underbrace{d_{ij}^y}_{=|y_i - y_j|} \geq \frac{1}{2}(h_i + h_j) \quad \forall i < j.$$

Since only one of the two inequalities must be satisfied for each pair  $i, j$  of modules, these constraints are commonly modeled using binary variables [64] or complementarity constraints [9]. The formulation in Takouda et al. [69] uses only two binary variables per pair of modules. The first variable,  $\sigma_{ij}$ , is used to decide the direction in which the non-overlap is enforced:

$$\sigma_{ij} = \begin{cases} +1 & \text{if } i \text{ and } j \text{ are separated along } x, \\ -1 & \text{if } i \text{ and } j \text{ are separated along } y, \end{cases} \quad (29.26)$$

Once this direction is selected, the variable  $\alpha_{ij}$  determines the relative position of modules  $i$  and  $j$  in that direction:

$$\alpha_{ij} = \begin{cases} +1 & \text{if } i \text{ precedes } j \text{ in the selected direction,} \\ -1 & \text{if } j \text{ precedes } i \text{ in the selected direction.} \end{cases} \quad (29.27)$$

The following inequalities enforce the separation of modules  $i$  and  $j$  in the  $x$  direction:

$$d_{ij}^x \geq \frac{1}{2}(w_i + w_j) - \frac{1}{2}(1 - \sigma_{ij})Q_{ij}^x \quad (29.28)$$

$$d_{ij}^x - 2S_{ij}^x = x_j - x_i \quad (29.29)$$

$$0 \leq S_{ij}^x \quad (29.30)$$

$$S_{ij}^x \leq \frac{1}{2} \left[ (1 - \sigma_{ij}) + \frac{1}{2}(1 + \sigma_{ij})(1 - \alpha_{ij}) \right] U_{ij}^x \quad (29.31)$$

$$0 \leq S_{ij}^x + (x_j - x_i) \quad (29.32)$$

$$S_{ij}^x + (x_j - x_i) \leq \frac{1}{2} \left[ (1 - \sigma_{ij}) + \frac{1}{2}(1 + \sigma_{ij})(1 + \alpha_{ij}) \right] U_{ij}^x \quad (29.33)$$

where  $Q_{ij}^x = \min\{w_F, w_i^{\max} + w_j^{\max}\}$  and  $U_{ij}^x = w_F - \frac{1}{2}w_i^{\min} - \frac{1}{2}w_j^{\min}$ . The constraints also compute the  $x$  component  $d_{ij}^x$  of the pairwise distance  $d_{ij}$ . A similar set of constraints can be used for the  $y$  direction of separation.

### 29.3.1.5 Complete Formulation

To complete the formulation, we model the rectilinear distances between a module  $i$  and an I/O pad  $p$ :

$$d_{ip}^r = \underbrace{|x_i - x_p|}_{d_{ip}^x} + \underbrace{|y_i - y_p|}_{d_{ip}^y},$$

where  $x_p$  and  $y_p$  are given. Since either  $|x_p| \geq w_F/2$  or  $|y_p| \geq h_F/2$ , we have two cases. If  $|x_p| \geq w_F/2$ :

$$d_{ip}^x = x_p - x_i \text{ or } d_{ip}^x = x_i - x_p \quad (\text{whichever is positive}) \quad (29.34)$$

$$d_{ip}^y \geq y_p - y_i \quad (29.35)$$

$$d_{ip}^y \geq y_i - y_p. \quad (29.36)$$

The case  $|y_p| \geq h_F/2$  is handled similarly.

The result is a mixed-integer SOCP-SDP formulation for the VLSI macrocell floorplanning problem:

$$\begin{aligned} & \min \sum_{i<j}^N c_{ij}(d_{ij}^x + d_{ij}^y) + \sum_{i=1}^N \sum_{p=1}^{N_p} \gamma_{ip}(d_{ip}^x + d_{ip}^y) \\ \text{s.t. Areas: } & (29.20) \quad \forall i \\ \text{Fit-in-the-chip: } & (29.21), (29.22) \quad \forall i \\ \text{Aspect ratios: } & (29.25) \quad \forall i \\ \text{Non-overlap \& distances (module to module): } & (29.28) - (29.33) \\ & \text{and similar constraints for the } y \text{ direction} \quad \forall i < j \\ \text{Distances (module to pad): } & (29.34) - (29.36) \forall i, p \\ 0 \leq w_i \leq w_F \text{ and } 0 \leq h_i \leq h_F & \quad \forall i \\ d_{ij}^x, d_{ij}^y, d_{ip}^x, d_{ip}^y, S_{ij}^x, S_{ij}^y, x_i, y_i \in \mathbb{R} & \quad \forall i < j \\ \sigma_{ij}, \alpha_{ij} \in \{-1, 1\} & \quad \forall i < j. \end{aligned}$$

It is of course straightforward to alternatively measure the distances using the Euclidean distance by using a second-order cone constraint. We focus on the rectilinear distance because it is the measure of interest in the context of floorplanning.

Additional inequalities can be used to slightly improve this formulation. We refer the reader to [69] for details.

### 29.3.2 Deriving an SDP Relaxation

First we expand the non-overlap constraints:

$$S_{ij}^x \leq \frac{1}{4} (3 - \sigma_{ij} - \alpha_{ij} - \sigma_{ij}\alpha_{ij}) U_{ij}^x, \quad (29.37)$$

$$S_{ij}^x + (x_j - x_i) \leq \frac{1}{4} (3 - \sigma_{ij} + \alpha_{ij} + \sigma_{ij}\alpha_{ij}) U_{ij}^x, \quad (29.38)$$

$$S_{ij}^y \leq \frac{1}{4} (3 + \sigma_{ij} - \alpha_{ij} + \sigma_{ij}\alpha_{ij}) U_{ij}^y, \quad (29.39)$$

$$S_{ij}^y + (y_j - y_i) \leq \frac{1}{4} (3 + \sigma_{ij} + \alpha_{ij} - \sigma_{ij}\alpha_{ij}) U_{ij}^y. \quad (29.40)$$

Thus we need an SDP relaxation that allows us to linearize the products of sigmas and alphas in these expressions.

We also ensure that our formulation satisfies the two-dimensional version of the transitivity property (29.12) for the SRFLP. We state it as:

If modules  $i, j, k$  are separated in the same direction,  
and if in that direction  $i$  precedes  $j$  and  $j$  precedes  $k$ ,  
then  $i$  precedes  $k$  in that direction.

Since

- The modules  $i, j, k$  are separated in the same direction if and only if  $\sigma_{ij} = \sigma_{jk} = \sigma_{ik}$ , and
- The transitivity property in the selected direction is then :  $\alpha_{ij} = \alpha_{jk} \Rightarrow \alpha_{ij} = \alpha_{ik}$ ,

the two-dimensional transitivity property is stated as

$$\begin{pmatrix} \sigma_{ij} = \sigma_{jk} = \sigma_{ik} \\ \text{and} \\ \alpha_{ij} = \alpha_{jk} \end{pmatrix} \Rightarrow \alpha_{ij} = \alpha_{ik}. \quad (29.41)$$

**Proposition 29.2.** [69] A sufficient condition to enforce (29.41) is

$$(\sigma_{ij} + \sigma_{jk})(\sigma_{ij} + \sigma_{ik})(\alpha_{ij} + \alpha_{jk})(\alpha_{ij} - \alpha_{ik}) = 0 \quad (29.42)$$

for all  $i < j < k$ .

Expanding (29.42), we obtain:

$$\begin{aligned} 1 &= \alpha_{ij}\alpha_{ik} - \alpha_{ij}\alpha_{jk} + \alpha_{ik}\alpha_{jk} \\ &\quad - \sigma_{ij}\sigma_{ik} - \sigma_{ij}\sigma_{jk} - \sigma_{ik}\sigma_{jk} \\ &\quad + \sigma_{ij}\sigma_{ik}\alpha_{ij}\alpha_{ik} - \sigma_{ij}\sigma_{ik}\alpha_{ij}\alpha_{jk} + \sigma_{ij}\sigma_{ik}\alpha_{ik}\alpha_{jk} \\ &\quad + \sigma_{ij}\sigma_{jk}\alpha_{ij}\alpha_{ik} - \sigma_{ij}\sigma_{jk}\alpha_{ij}\alpha_{jk} + \sigma_{ij}\sigma_{jk}\alpha_{ik}\alpha_{jk} \\ &\quad + \sigma_{ik}\sigma_{jk}\alpha_{ij}\alpha_{ik} - \sigma_{ik}\sigma_{jk}\alpha_{ij}\alpha_{jk} + \sigma_{ik}\sigma_{jk}\alpha_{ik}\alpha_{jk}, \end{aligned} \quad (29.43)$$

for all  $i < j < k$ . The SDP relaxation must allow us to linearize the products in these expressions as well.

To define the SDP matrix variable, we introduce the vector  $\xi$  of  $1 + 2\binom{N}{2} + 2\binom{\binom{N}{2}}{2}$  binary variables:

$$\xi = (1, \sigma_{12}, \dots, \sigma_{N-1,N}, \alpha_{12}, \dots, \alpha_{N-1,N}, \sigma_{12}\sigma_{13}, \dots, \sigma_{N-2,N}\sigma_{N-1,N}, \alpha_{12}\alpha_{13}, \dots, \alpha_{N-2,N}\alpha_{N-1,N})^T$$

containing all the binary variables and their required monomials. The SDP rank-one formulation and subsequent relaxation are constructed as in Sect. 29.2.5 with the vector  $\xi$  replacing the vector  $v$ .

**Table 29.2** Results on MCNC benchmark problems

Circuit	Number of modules	Number of pads	Aspect ratio	Lower bound	CPU time (sec)
Apte	9	73	2	3135.9	891
			3	3021.6	789
			5	2918.5	848
			8	2848.8	815
			10	2847.7	793
Xerox	10	2	2	2434.0	1,665
			3	2051.0	2,290
			5	1539.8	2,930
			8	1217.1	2,835
			10	1153.1	2,721
Hp	11	45	2	976.98	8,156
			3	893.25	8,230
			5	822.11	8,294
			8	783.80	7,840
			10	773.23	7,855

### 29.3.3 Computational Performance of the SDP Relaxation

The MCNC benchmarks are a well-known collection of benchmark problems used by VLSI researchers. Takouda et al. [69] applied the SDP relaxation to three problems from the MCNC benchmark: apte, xerox and hp. Although these are the smallest instances in this benchmark, and this benchmark is now several years old, these problems are quite large from the facility layout perspective, and hence already challenging.

The bounds for varying aspect ratios are reported in Table 29.2. To give a sense of the computational effort required to solve the relaxations, we report in Table 29.2 the CPU time taken using CSDP [14] on a 2.0 GHz Dual Opteron with 16Gb of RAM. Finally, to convey a sense of the quality of the bounds, we compare them in Table 29.3 to the best reported solutions in the literature, namely those of Murata and Kuh [57]. We point out that the xerox instance is particularly challenging. This is true not only for the SDP relaxation but also for macrocell placement algorithms, and is due to the very small number of pads.

Overall, although this is the first time that such lower bounds were computed, we see that the relative gaps to the best known solutions are still large. Takouda et al. improved the bounds slightly by computing full levels of a branch-and-bound tree, but such an approach is of limited use because of the resulting high computational times. It is thus necessary to improve the strength of the SDP relaxations, and to be able to solve them for larger sizes. Current research is making good progress in these directions [1].

**Table 29.3** Global bounds versus Murata–Kuh solutions ( $\beta_i = 10$ )

Circuit	Bound	Solution	Gap (%)
Apte	2847.7	4353.5	34.6
Xerox	1153.1	4976.5	76.8
Hp	773.2	1779.8	56.6

## 29.4 Ongoing and Future Research

There is still much potential for the SDP-based approach in the areas of facility layout and floorplanning. First, the recent results of Hungerländer and Rendl [39] suggest that the potential of the SDP approach for the SRFLP is not yet fully developed. While it is generally accepted that memory requirements for SDP are higher than for LP, and that LP can exploit the sparsity in the SRFLP objective better than SDP, the SDP bounds stronger than the LP bounds. It would therefore be interesting to conduct a careful study of the computational limitations and future potential of LP and SDP approaches. Such a study should include an assessment of how the SDP rounding heuristic in Sect. 29.2.6 compares to other heuristics applied to the SRFLP in the literature.

Second, extensions of the conic optimization approach to other layout problems are still in their infancy. For two-dimensional layout, Adams [1] proposed a novel SDP relaxation that replaces the variables  $\sigma_{ij}$  and  $\alpha_{ij}$  described above with quaternary variables. Preliminary results indicate that this approach outperforms the model of Takouda et al. described above. Other important research directions that remain unexplored include the extension of the SDP models for SRFLP to multi-row problems with applications in the service industry and VLSI design. A second one is the application of SDP to three-dimensional layout problems for which current global optimization methods are of limited use, see e.g. [13] and the references therein. Thus, layout problems remain a very promising opportunity for the application of conic optimization techniques to obtain global optimal solutions.

**Acknowledgements** The authors gratefully acknowledge the support provided by the following institutions: The Alexander von Humboldt Foundation and the Natural Sciences and Engineering Research Council of Canada (first author), and the German Science Foundation (second author).

## References

1. Adams, E.C.: A semidefinite programming model for the facility layout problem. Master's thesis, University of Waterloo (2010)
2. Amaral, A.R.S.: On the exact solution of a facility layout problem. *Eur. J. Oper. Res.* **173**(2), 508–518 (2006)
3. Amaral, A.R.S.: An exact approach for the one-dimensional facility layout problem. *Oper. Res.* **56**(4), 1026–1033 (2008)
4. Amaral, A.R.S.: A new lower bound for the single row facility layout problem. *Discrete Appl. Math.* **157**(1), 183–190 (2009)

5. Amaral, A.R.S.: On the exact solution of a facility layout problem. *Discr. Appl. Math.* **157**(1), 183–190 (2009)
6. Amaral, A.R.S., Letchford, A.N.: A polyhedral approach to the single-row facility layout problem. Technical report, Department of Management Science, Lancaster University (March 2008)
7. Anjos, M.F., Kennings, A., Vannelli, A.: A semidefinite optimization approach for the single-row layout problem with unequal dimensions. *Discrete Optim.* **2**(2), 113–122 (2005)
8. Anjos, M.F., Vannelli, A.: Globally optimal solutions for large single-row facility layout problems. Technical report, University of Waterloo (2006)
9. Anjos, M.F., Vannelli, A.: A new mathematical-programming framework for facility-layout design. *INFORMS J. Comp.* **18**(1), 111–118 (2006)
10. Anjos, M.F., Vannelli, A.: Computing globally optimal solutions for single-row layout problems using semidefinite programming and cutting planes. *INFORMS J. Comp.* **20**(4), 611–617 (2008)
11. Anjos, M.F., Yen, G.: Provably near-optimal solutions for very large single-row facility layout problems. *Optim. Methods Softw.* **24**(4), 805–817 (2009)
12. Barahona, F., Mahjoub, A.R.: On the cut polytope. *Mathematical Programming* **36**, 157–173, 1986.
13. Bernardi, S.: A three-stage mathematical-programming method for the multi-floor facility layout problem. Master's thesis, University of Waterloo (2010)
14. Borchers, B.: CSDP, a C library for semidefinite programming. *Optim. Methods Softw.* **11/12**(1-4), 613–623 (1999)
15. Buchheim, C., Liers, F., Oswald, M.: Speeding up IP-based algorithms for constrained quadratic 0-1 optimization. *Mathematical Programming (Series B)* **124**(1-2), 513–535 (2010)
16. Buchheim, C., Wiegele, A., Zheng, L.: Exact algorithms for the quadratic linear ordering problem. *INFORMS J. on Computing* **2**(1), 168–177 (2010)
17. Caprara, A., Oswald, M., Reinelt, G., Schwarz, R., Traversi, E.: Optimal linear arrangements using betweenness variables. *Mathematical Programming (Series C)* **3**(3), 261–280 (2011)
18. Castillo, I., Sim, T.: A spring-embedding approach for the facility layout problem. *J. Oper. Res. Soc.* **55**, 73–81 (2004)
19. Castillo, I., Westerlund, J., Emet, S., Westerlund, T.: Optimization of block layout design problems with unequal areas: A comparison of MILP and MINLP optimizition methods. *Computers and Chemical Engineering* **30**(1), 54–69 (2005)
20. Castillo, I., Westerlund, T.: An  $\varepsilon$ -accurate model for optimal unequal-area block layout design. *Comput. Oper. Res.* **32**(3), 429–447 (2005)
21. Çela, E.: The Quadratic Assignment Problem. In: *Combinatorial Optimization*, vol. 1. Kluwer Academic Publishers, Dordrecht (1998)
22. Charon, I., Hudry, O.: An updated survey on the linear ordering problem for weighted or unweighted tournaments. *Ann. Oper. Res.* **175**, 107–158 (2010)
23. Christof, T., Oswald, M., Reinelt, G.: Consecutive ones and a betweenness problem in computational biology. In *Proceedings of the 6th Conference on Integer Programming and Combinatorial Optimization*, Springer-Verlag Lecture Notes in Computer Science 1412 (1998)
24. de Alvarenga, A.G., Negreiros-Gomes, F.J., Mestria, M.: Metaheuristic methods for a class of the facility layout problem. *J. Intell. Manuf.* **11**, 421–430 (2000)
25. De Simone, C.: The cut polytope and the boolean quadric polytope. *Discrete Mathematics* **79**, 71–75 (1989)
26. Deza, M.M., Laurent, M.: *Geometry of Cuts and Metrics*, vol. 15 of Algorithms and Combinatorics. Springer-Verlag, Berlin (1997)
27. Díaz, J., Petit, J., Serna, M.: A survey on graph layout problems. *ACM Computing Surveys* **34**, 313–356 (2002)
28. Foulds, L.R.: *Graph Theory Applications*. Springer-Verlag, New York (1991)
29. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.* **42**(6), 1115–1145 (1995)

30. Grötschel, M., Jünger, M., Reinelt, G.: Facets of the linear ordering polytope. *Mathematical Programming* **33**, 43–60 (1985)
31. Hall, K.M.: An  $r$ -dimensional quadratic placement algorithm. *Management Sciences* **17**, 219–229 (1970)
32. Hammer, P.L.: Some network flow problems solved with pseudo-boolean programming. *Operations Research* **13**, 388–399 (1965)
33. Helmberg, C., Kiwiel, K.C.: A spectral bundle method with bounds. *Math. Program.* **93**(2, Ser. A), 173–194 (2002)
34. Helmberg, C., Rendl, F.: A spectral bundle method for semidefinite programming. *SIAM J. Optim.* **10**(3), 673–696 (2000)
35. Heragu, S.S.: Facilities Design. iUniverse, second edition (2006)
36. Heragu, S.S., Alfa, A.S.: Experimental analysis of simulated annealing based algorithms for the layout problem. *European J. Oper. Res.* **57**(2), 190–202 (1992)
37. Heragu, S.S., Kusiak, A.: Machine layout problem in flexible manufacturing systems. *Oper. Res.* **36**(2), 258–268 (1988)
38. Heragu, S.S., Kusiak, A.: Efficient models for the facility layout problem. *European J. Oper. Res.* **53**, 1–13 (1991)
39. Hungerländer, P., Rendl, F.: Semidefinite relaxations of ordering problems. Technical report, Alpen-Adria-Universität Klagenfurt (August 2010)
40. Jankovits, I.: An improved convex optimization model for two-dimensional facility layout. Master's thesis, University of Waterloo (2006)
41. Karp, R.M., Held, M.: Finite-state processes and dynamic programming. *SIAM J. Appl. Math.* **15**, 693–718 (1967)
42. Kim, S., Kojima, M., Yamashita, M.: Second order cone programming relaxation of a positive semidefinite constraint. *Optim. Methods Softw.* **18**(5), 535–541 (2003)
43. Koopmans, T.C., Beckmann, M.: Assignment problems and the location of economic activities. *Econometrica* **25**, 53–76 (1957)
44. Kumar, K.R., Hadjinicola, G.C., Lin, T.: A heuristic procedure for the single-row facility layout problem. *European J. Oper. Res.* **87**(1), 65–73 (1995)
45. Lewis, M., Alidaee, B., Glover, F., Kochenberger, G.: A note on  $xQx$  as a modelling and solution framework for the linear ordering problem. *International Journal of Operational Research* **5**(2), 152–162 (2009)
46. Liers, F., Jünger, M., Reinelt, G., Rinaldi, G.: Computing Exact Ground States of Hard Ising Spin Glass Problems by Branch-and-Cut, pp. 47–68. *New Optimization Algorithms in Physics*. Wiley-VCH (2004)
47. Liu, W., Vannelli, A.: Generating lower bounds for the linear arrangement problem. *Discrete Appl. Math.* **59**(2), 137–151 (1995)
48. Lovász, L., Schrijver, A.: Cones of matrices and set-functions and 0-1 optimization. *SIAM J. Optim.* **1**, 166–190 (1991)
49. Love, R.F., Wong, J.Y.: On solving a one-dimensional space allocation problem with integer programming. *INFOR* **14**(2), 139–143 (1976)
50. Luo, C., Anjos, M.F., Vannelli, A.: A nonlinear optimization methodology for VLSI fixed-outline floorplanning. *J. Comb. Optim.* **16**(4), 378–401 (2008)
51. Mavridou, T.D., Pardalos, P.M.: Simulated annealing and genetic algorithms for the facility layout problem: a survey. *Comput. Optim. Appl.*, **7**(1), 111–126 (1997)
52. Meller, R.D., Chen, W., Sherali, H.D.: Applying the sequence-pair representation to optimal facility layout designs. *Oper. Res. Lett.* **35**(5), 651–659 (2007)
53. Meller, R.D., Gau, K.Y.: The facility layout problem: recent and emerging trends and perspectives. *Journal of Manufacturing Systems* **15**, 351–366 (1996)
54. Meller, R.D., Narayanan, V., Vance, P.H.: Optimal facility layout design. *Oper. Res. Lett.* **23**(3–5), 117–127 (1998)
55. Montreuil, B.: A modelling framework for integrating layout design and flow network design. In: White, J.A., Pence, I.W. (eds.) *Progress in Material Handling and Logistics*, vol. 2, pp. 95–116. Springer-Verlag (1991)

56. Murata, H., Fujiyoshi, K., Nakatake, S., Kajitani, Y.: VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems* **15**(12), 1518–1524 (1996)
57. Murata, H., Kuh, E.S.: Sequence-pair based placement method for hard/soft/pre-placed modules. *Proceedings of the International Symposium on Physical Design*, pp. 167–172 (1998)
58. Picard, J.-C., Queyranne, M.: On the one-dimensional space allocation problem. *Oper. Res.* **29**(2), 371–391 (1981)
59. Reinelt, G.: *The Linear Ordering Problem: Algorithms and Applications*. Heldermann Verlag (1985)
60. Rendl, F., Rinaldi, G., Wiegele, A.: A branch and bound algorithm for Max-Cut based on combining semidefinite and polyhedral relaxations. In *Integer programming and combinatorial optimization*, vol. 4513 of *Lecture Notes in Computer Science*, pp. 295–309. Springer Verlag, Berlin (2007)
61. Romero, D., Sánchez-Flores, A.: Methods for the one-dimensional space allocation problem. *Comput. Oper. Res.* **17**(5), 465–473 (1990)
62. Sait, S.M., Youssef, H.: *VLSI physical design automation: theory and practice*. IEEE Press, New York, USA (1995)
63. Sanjeevi, S., Kianfar, K.: A polyhedral study of triplet formulation for single row facility layout problem. *Discrete Appl. Math.* **158**(16), 1861–1867 (2010)
64. Sherali, H.D., Fraticelli, B.M.P., Meller, R.D.: Enhanced model formulation for optimal facility layout. *Oper. Res.* **51**(4), 629–644 (2003)
65. Simmons, D.M.: One-dimensional space allocation: An ordering algorithm. *Oper. Res.* **17**, 812–826 (1969)
66. Simmons, D.M.: A further note on one-dimensional space allocation. *Oper. Res.* **19**, p. 249 (1971)
67. Singh, S.P., Sharma, R.R.K.: A review of different approaches to the facility layout problems. *Intl J. Adv. Manuf. Tech.* **30**(5–6), 425–433 (2006)
68. Suryanarayanan, J.K., Golden, B.L., Wang, Q.: A new heuristic for the linear placement problem. *Computers & Operations Research* **18**(3), 255–262 (1991)
69. Takouda, P.L., Anjos, M.F., Vannelli, A.: Global lower bounds for the VLSI macrocell floorplanning problem using semidefinite optimization. In *Proceedings of IWSOC 2005*, pp. 275–280 (2005)
70. Wong, D.F., Liu, C.L.: A new algorithm for floorplan design. In: *Proc. of ACM/IEEE Design Automation Conf.*, pp. 101–107 (1986)
71. Xie, W., Sahinidis, N.V.: A branch-and-bound algorithm for the continuous facility layout problem. *Comput. & Chem. Eng.* **32**, 1016–1028 (2008)

# Chapter 30

## Euclidean Distance Matrices and Applications

Nathan Krislock and Henry Wolkowicz

### 30.1 Introduction

Over the past decade, Euclidean distance matrices, or EDMs, have been receiving increased attention for two main reasons. The first reason is that the many applications of EDMs, such as molecular conformation in bioinformatics, dimensionality reduction in machine learning and statistics, and especially the problem of wireless sensor network localization, have all become very active areas of research. The second reason for this increased interest is the close connection between EDMs and semidefinite matrices. Our recent ability to solve semidefinite programs, SDPs, efficiently means we can now also solve many problems involving EDMs efficiently.

#### 30.1.1 Background

##### 30.1.1.1 Distance Geometry and Euclidean Distance Matrices

Two foundational papers in the area of Euclidean distance matrices are [104] and [119]. The topic was further developed with the series of papers [63–65], followed by [43, 54]. For papers on the Euclidean distance matrix completion problem and the related semidefinite completion problem, see the classic paper on semidefinite completion [67], and follow-up papers [19] and [78]; also see [87] on the topic of the complexity of these completion problems. More on the topic of uniqueness of Euclidean distance matrix completions can be found in the papers [8, 9]. The cone of Euclidean distance matrices and its geometry is described in, for

---

N. Krislock (✉) • H. Wolkowicz  
Department of Combinatorics and Optimization, University of Waterloo,  
Waterloo, ON, Canada N2L 3G1  
e-mail: [nkbkrislock@uwaterloo.ca](mailto:nkbkrislock@uwaterloo.ca); [hwolkowicz@uwaterloo.ca](mailto:hwolkowicz@uwaterloo.ca)

example, [11, 59, 71, 110, 111]. Using semidefinite optimization to solve Euclidean distance matrix problems is studied in [2, 4]. Further theoretical results are given in [10, 13]. Books and survey papers containing a treatment of Euclidean distance matrices include, for example, [31, 44, 86], and most recently [3]. The topic of rank minimization for Euclidean distance matrix problems is discussed in, for example, [34, 35, 55, 56, 98, 99].

### 30.1.1.2 Graph Realization and Graph Rigidity

The complexity of graph realization in a fixed dimension was determined to be NP-hard by [102, 118]. For studies on graph rigidity, see, for example, [6, 7, 12, 23, 24, 39, 73, 74, 76], and the references therein. Graph rigidity for sensor network localization is studied as graph rigidity with some nodes being grounded or anchored; see, for example, [53, 108]. Semidefinite optimization techniques have also been applied to graph realization and graph rigidity problems; see, for example, [20, 106, 107].

### 30.1.1.3 Sensor Network Localization

While semidefinite relaxations were discussed earlier in [4] for Euclidean distance matrix problems, the first semidefinite relaxations specialized for the sensor network localization problem were proposed by [47]. The paper by [28] followed with what is now called the Biswas–Ye semidefinite relaxation of sensor network localization problem. As problems with only a few hundred sensors could be solved directly using the Biswas–Ye relaxation, [77] and [36] proposed the scalable SpaseLoc semidefinite-based build-up algorithm which solves small subproblems using the Biswas–Ye semidefinite relaxation, locating a few sensors at a time. In the follow-up paper [26] propose regularization and refinement techniques for handling noisy problems using the Biswas–Ye semidefinite relaxation. In order to handle larger problems, a distributed method was proposed in [29] which clusters points together in small groups, solves the smaller subproblems, then stitches the clusters together; see also the PhD thesis [25]. To allow the solution of larger problems, [113] proposed further relaxations of the sensor network localization problem in which they do not insist that the full  $n$ -by- $n$  matrix of the Biswas–Ye relaxation be positive semidefinite, but rather only that certain submatrices of this matrix be positive semidefinite; the most successful of these further relaxations is the so-called edge-based SDP relaxation, or ESDP. A noise-aware robust version of the ESDP relaxation called  $\rho$ -ESDP is proposed by [96] and which is solved with their Log-barrier Penalty Coordinate Gradient Descent (LPCGD) method. Using techniques from [57], it is shown in [80] how to form an equivalent sparse version of the full Biswas–Ye relaxation, called SFSDP. The sparsity in the SFSDP formulation can then be exploited by a semidefinite optimization solver, allowing the solution of noisy instances of the sensor network localization problem with up to 18,000 sensors and 2,000 anchors to high accuracy in under ten minutes; see [81]. Most recently,

it was shown in [83] how to use facial reduction to solve a semidefinite relaxation of the sensor network localization problem; the resulting algorithm is able to solve noiseless problems with up to 100,000 sensors and 4 anchors to high accuracy in under six minutes on a laptop computer. The connection between the sensor network localization problem and the Euclidean distance matrix problem is described in [3]. In particular, the connection uses facial reduction based on the clique of anchors.

Other relaxations have also been studied; [112] considers a second-order cone (SOC) relaxation of the sensor network localization problem, while [95] studies the sum-of-squares (SOS) relaxation of this problem.

For more applied approaches and general heuristics, see, for example, [32, 33, 37, 40, 90, 91, 94, 101, 109, 117]. A older survey paper on wireless sensor networks is [1]; for a recent book on wireless ad hoc and sensor networks, see [89].

The complexity of the sensor network localization problem is discussed in [16, 17]. References for the single sensor localization problem are, for example, [21, 22].

### 30.1.1.4 Molecular Conformation

An early algorithmic treatment of molecular conformation is [69] in which they give their bound embedding algorithm EMBED. This paper was then followed by the book [42]; a review paper [41] provides an update three years after the publication of this book. A personal historical perspective is given in [70].

Other algorithmic developments followed, including: a divide-and-conquer algorithm called ABBIE based on identifying rigid substructures [72]; an alternating projection approach [58]; a global smoothing continuation code called DGSOL [92, 93]; a geometric build-up algorithm [48, 49, 115, 116]; an extended recursive geometric build-up algorithm [50]; a difference of convex functions (d.c.) optimization algorithm [15]; a method based on rank-reducing perturbations of the distance matrix that maintain desired structures [52]; an algorithm for solving a distance matrix based, large-scale, bound constrained, non-convex optimization problem called STRAINMIN [68].

Recently, semidefinite optimization approaches to the molecular conformation problem have been studied in [25, 27, 88].

## 30.1.2 Outline

We begin in Sect. 30.2 by discussing some preliminaries and introducing notation. In Sect. 30.3, we explain the close connection to semidefinite matrices and the many recent results arising from this special relationship. In Sect. 30.5, we will look at some popular applications, and we especially focus on the problem of sensor network localization (SNL).

## 30.2 Preliminaries

We let  $\mathcal{S}^n$  be the space of  $n \times n$  real symmetric matrices. A *Euclidean distance matrix* (EDM) is a matrix  $D$  for which

$$\exists p_1, \dots, p_n \in \mathbb{R}^r, \text{ such that } D_{ij} = \|p_i - p_j\|_2^2, \quad \forall i, j = 1, \dots, n. \quad (30.1)$$

The set of Euclidean distance matrices is denoted  $\mathcal{E}^n$ . If  $D$  is an EDM, then the smallest integer  $r$  for which condition (30.1) is possible is called the *embedding dimension* of  $D$ , and is denoted  $\text{embdim}(D)$ .

### 30.2.1 Further Notation

The adjoint of a linear transformation  $T$  is denoted  $T^*$  and satisfies  $\langle Tx, y \rangle = \langle x, T^*y \rangle, \forall x, y$ . For a given matrix  $M$  and vector  $v$ , we let  $\text{diag}(M)$  denote the vector formed from the diagonal of  $M$ . The adjoint  $\text{Diag}(v) = \text{diag}^*(v)$  is the diagonal matrix formed with  $v$  as its diagonal. For a given symmetric matrix  $B$ , we let  $B[\alpha]$  denote the principal submatrix formed using the rows/columns from the index set  $\alpha$ .

The cone of symmetric positive semidefinite (resp. definite) matrices is denoted by  $\mathcal{S}_+^n$  (resp.  $\mathcal{S}_{++}^n$ ). The cone  $\mathcal{S}_+^n$  is closed and convex and induces the Löwner partial order:

$$A \succeq B \text{ (resp. } A > B\text{),} \quad \text{if } A - B \in \mathcal{S}_+^n \text{ (resp. } A - B \in \mathcal{S}_{++}^n\text{).}$$

A convex cone  $F \subseteq K$  is a *face of the cone  $K$* , denoted  $F \trianglelefteq K$ , if

$$\left( x, y \in K, \frac{1}{2}(x+y) \in F \right) \implies (x, y \in F).$$

If  $F \trianglelefteq K$ , but is not equal to  $K$ , we write  $F \lhd K$ . If  $\{0\} \neq F \lhd K$ , then  $F$  is a *proper face* of  $K$ . For  $S \subseteq K$ , we let  $\text{face}(S)$  denote the smallest face of  $K$  that contains  $S$ . If  $F \trianglelefteq K$ , the *conjugate face* of  $F$  is  $F^c := F^\perp \cap K^*$ , where  $K^* := \{x : \langle x, y \rangle \geq 0, \forall y \in K\}$  is the *dual cone* of the cone  $K$ ; in fact it is easy to show that  $F^c \trianglelefteq K^*$  and that if  $\phi \in F^c$ , then  $F \trianglelefteq K \cap \{\phi\}^\perp$  (see, for example, [82]). A face  $F \trianglelefteq K$  is an *exposed face* if it is the intersection of  $K$  with a hyperplane. It is well known that  $\mathcal{S}_+^n$  is *facially exposed*: every face  $F \trianglelefteq \mathcal{S}_+^n$  is exposed. For more on faces of convex cones, the interested reader is encouraged to refer to [97, 100, 103].

### 30.3 Euclidean Distance Matrices and Semidefinite Matrices

The connection between EDMs and semidefinite matrices is well known. This has been studied at length in, e.g., [78, 85, 86]. There is a natural relationship between the sets  $\mathcal{S}_+^n$  and  $\mathcal{E}^n$ . Suppose that  $D \in \mathcal{E}^n$  is realized by the points  $p_1, \dots, p_n \in \mathbb{R}^r$ . Let

$$P := \begin{bmatrix} p_1^T \\ \vdots \\ p_n^T \end{bmatrix} \in \mathbb{R}^{n \times r} \quad \text{and} \quad Y := PP^T = (p_i^T p_j)_{i,j=1}^n.$$

The matrix  $Y = PP^T$  is known as the *Gram matrix* of the points  $p_1, \dots, p_n$ . Then,  $\forall i, j \in \{1, \dots, n\}$ , we have

$$\begin{aligned} D_{ij} &= \|p_i - p_j\|_2^2 \\ &= p_i^T p_i + p_j^T p_j - 2p_i^T p_j \\ &= Y_{ii} + Y_{jj} - 2Y_{ij}. \end{aligned}$$

Therefore,  $D = \mathcal{K}(Y)$ , where  $\mathcal{K}: \mathcal{S}^n \rightarrow \mathcal{S}^n$  is the linear operator<sup>1</sup> defined as

$$\mathcal{K}(Y)_{ij} := Y_{ii} + Y_{jj} - 2Y_{ij}, \quad \text{for } i, j = 1, \dots, n.$$

Equivalently, we can define  $\mathcal{K}$  by

$$\mathcal{K}(Y) := \text{diag}(Y)e^T + e\text{diag}(Y)^T - 2Y, \tag{30.2}$$

where  $e \in \mathbb{R}^n$  is the vector of all ones. From this simple observation, we can see that  $\mathcal{K}$  maps the cone of semidefinite matrices,  $\mathcal{S}_+^n$ , onto  $\mathcal{E}^n$ . That is,  $\mathcal{K}(\mathcal{S}_+^n) = \mathcal{E}^n$ . In addition, since  $\mathcal{S}_+^n$  is a convex cone, we immediately get that  $\mathcal{E}^n$  is a convex cone.

#### 30.3.1 Mappings Between EDM and SDP

There are several linear transformations that map between  $\mathcal{E}^n$  and the cone of semidefinite matrices. We first present some useful properties of  $\mathcal{K}$  in (30.2) and related transformations and their adjoints follow; see, e.g., [83]. For  $Y \in \mathcal{S}^n$  we

---

<sup>1</sup>Early appearances of this linear operator are in [104, 119]. However, the use of the notation  $\mathcal{K}$  for this linear operator dates back to [43] wherein  $\kappa$  was used due to the fact that the formula for  $\mathcal{K}$  is basically the *cosine law* ( $c^2 = a^2 + b^2 - 2ab \cos(\gamma)$ ). Later on, in [78],  $K$  was used to denote this linear operator.

let  $\mathcal{D}_e(Y) := \text{diag}(Y)e^T + e \text{ diag}(Y)^T$ ; by abuse of notation, we also let  $\mathcal{D}_e(y) := ye^T + ey^T$ , for  $y \in \mathbb{R}^n$ . Then, our main operator of interest is

$$\mathcal{K}(Y) := \mathcal{D}_e(Y) - 2Y.$$

The adjoints are

$$\mathcal{D}_e^*(D) = 2\text{Diag}(De), \quad \mathcal{K}^*(D) = 2(\text{Diag}(De) - D).$$

We also have an explicit representation for the Moore–Penrose generalized inverse:

$$\boxed{\mathcal{K}^\dagger(D) = -\frac{1}{2}J \text{ offDiag}(D)J}$$

where  $J := I - \frac{1}{n}ee^T$ ,  $\text{offDiag}(D) := D - \text{Diag}(\text{diag}(D))$ . In addition,

$$\mathcal{S}_H^n := \{D \in \mathcal{S}^n : \text{diag}(D) = 0\} \quad \mathcal{K}\mathcal{K}^\dagger(D) = \text{offDiag}(D)$$

$$\mathcal{S}_C^n := \{Y \in \mathcal{S}^n : Ye = 0\} \quad \mathcal{K}^\dagger\mathcal{K}(Y) = JYJ$$

$\mathcal{S}_H^n$  is called the *hollow subspace*, while  $\mathcal{S}_C^n$  is called the *centered subspace*.

$$\text{range}(\mathcal{K}) = \mathcal{S}_H^n \quad \text{null}(\mathcal{K}^\dagger) = \text{range}(\text{Diag})$$

$$\text{range}(\mathcal{K}^\dagger) = \mathcal{S}_C^n \quad \text{null}(\mathcal{K}) = \text{range}(\mathcal{D}_e)$$

$$\mathcal{K}(\mathcal{S}_C^n) = \mathcal{S}_H^n \quad \mathcal{K}^\dagger(\mathcal{S}_H^n) = \mathcal{S}_C^n$$

$$\mathcal{K}(\mathcal{S}_+^n \cap \mathcal{S}_C^n) = \mathcal{E}^n \quad \mathcal{K}^\dagger(\mathcal{E}^n) = \mathcal{S}_+^n \cap \mathcal{S}_C^n$$

$$\text{embdim}(D) = \text{rank } \mathcal{K}^\dagger(D), \quad \text{for } D \in \mathcal{E}^n$$

$$\|\mathcal{K}\|_F := \max_{0 \neq Y \in \mathcal{S}^n} \frac{\|\mathcal{K}(Y)\|_F}{\|Y\|_F} = 2\sqrt{n}$$

We let  $\mathcal{T} := \mathcal{K}^\dagger$ . Then  $\mathcal{K}$  and  $\mathcal{T}$  map between the centered subspace,  $\mathcal{S}_C^n$ , and the hollow subspace,  $\mathcal{S}_H^n$ . Since  $\text{int}(\mathcal{S}_C^n \cap \mathcal{S}_+^n) = \emptyset$ , we can have problems with constraint qualifications and unbounded optimal sets. To avoid this ([2,4]), we define  $\mathcal{K}_V: \mathcal{S}^{n-1} \rightarrow \mathcal{S}^n$  by

$$\mathcal{K}_V(X) := \mathcal{K}(VXV^T), \tag{30.3}$$

where  $V \in \mathbb{R}^{n \times (n-1)}$  is full column rank and satisfies  $V^T e = 0$ . Then  $\mathcal{K}_V(\mathcal{S}_+^{n-1}) = \mathcal{E}^n$ . And,  $\mathcal{K}_V(X) = D \in \mathcal{E}^n$  implies that  $VXV^T$  is the corresponding Gram matrix.

Alternatively, see [2], we can use  $\mathcal{L}: \mathcal{S}^{n-1} \rightarrow \mathcal{S}^n$

$$\mathcal{L}(X) := \begin{bmatrix} 0 & \text{diag}(X)^T \\ \text{diag}(X) & \mathcal{K}(X) \end{bmatrix}. \quad (30.4)$$

And, as with  $\mathcal{K}_V$ , we get  $\mathcal{L}(\mathcal{S}_+^{n-1}) = \mathcal{E}^n$ .

### 30.3.2 Properties of $\mathcal{K}$

We now include further useful properties of the linear map  $\mathcal{K}$ .

#### 30.3.2.1 Translational Invariance and the Null Space of $\mathcal{K}$

The null space of  $\mathcal{K}$  is closely related to the translational invariance of distances between a set of points. Suppose that  $P \in \mathbb{R}^{n \times r}$  and that

$$\hat{P} = P + ev^T, \quad \text{for some } v \in \mathbb{R}^r;$$

that is,  $\hat{P}$  is the matrix formed by translating every row of  $P$  by the vector  $v$ . Clearly,  $P$  and  $\hat{P}$  generate the same Euclidean distance matrix, so we have  $\mathcal{K}(PP^T) = \mathcal{K}(\hat{P}\hat{P}^T)$ . Note that

$$\begin{aligned} \hat{P}\hat{P}^T &= PP^T + Pv e^T + ev^T P + ev^T ve^T \\ &= PP^T + \mathcal{D}_e(y), \end{aligned}$$

where  $y := Pv + \frac{v^T v}{2}e$ . Thus, we have

$$0 = \mathcal{K}(\hat{P}\hat{P}^T - PP^T) = \mathcal{K}(\mathcal{D}_e(y)),$$

so  $\mathcal{D}_e(y) \in \text{null}(\mathcal{K})$ . Thus, if there are no *anchors* among the points, then we do not have to worry about translations of the set of points.

#### 30.3.2.2 Rotational Invariance of the Gram Matrix

Suppose that  $P \in \mathbb{R}^{n \times r}$  and that

$$\hat{P} = PQ, \quad \text{for some } Q \in \mathbb{R}^{r \times r} \text{ orthogonal};$$

that is,  $\hat{P}$  is the matrix formed by rotating/reflecting each row of  $P$  by the same orthogonal transformation. Again, we clearly have that  $P$  and  $\hat{P}$  generate the same Euclidean distance matrix, but we can say more. If  $Y$  is the Gram matrix of  $P$  and  $\hat{Y}$  is the Gram matrix of  $\hat{P}$ , then

$$\hat{Y} = \hat{P}\hat{P}^T = PQQ^TP^T = PP^T = Y.$$

Therefore, we have that the Gram matrix is invariant under orthogonal transformations of the points. Thus, when using a semidefinite matrix  $Y$  to represent a Euclidean distance matrix  $D$  with  $D = \mathcal{K}(Y)$ , orthogonal transformations will not affect  $Y$  nor  $D$ .

### 30.3.2.3 The Maps $\mathcal{K}$ and $\mathcal{K}^\dagger$ as Bijections

Consider  $\mathcal{K}$  and  $\mathcal{K}^\dagger$  restricted to the subspaces  $\mathcal{S}_C^n$  and  $\mathcal{S}_H^n$ , respectively. Then, the above expressions for the ranges implies that the map  $\mathcal{K}: \mathcal{S}_C^n \rightarrow \mathcal{S}_H^n$  is a bijection and  $\mathcal{K}^\dagger: \mathcal{S}_H^n \rightarrow \mathcal{S}_C^n$  is its inverse.

If we consider  $\mathcal{K}$  and  $\mathcal{K}^\dagger$  restricted to the convex cones  $\mathcal{S}_+^n \cap \mathcal{S}_C^n$  and  $\mathcal{E}^n$ , respectively, then the map  $\mathcal{K}: \mathcal{S}_+^n \cap \mathcal{S}_C^n \rightarrow \mathcal{E}^n$  is a bijection and  $\mathcal{K}^\dagger: \mathcal{E}^n \rightarrow \mathcal{S}_+^n \cap \mathcal{S}_C^n$  is its inverse. Note that we could use a rotation and replace the face  $\mathcal{S}_+^n \cap \mathcal{S}_C^n \triangleleft \mathcal{S}_+^n$ , as is done above with the  $\mathcal{K}_V$  linear transformation defined in (30.3).

### 30.3.2.4 Embedding Dimension and a Theorem of Schoenberg

We now give the following much celebrated theorem of Schoenberg [104] (also found in the later paper by Young and Householder [119]) that provides a method for testing if a matrix is a Euclidean distance matrix and a method for determining the embedding dimension of a Euclidean distance matrix; see also [4].

**Theorem 30.1** ([104, 119]). *A matrix  $D \in \mathcal{S}_H^n$  is a Euclidean distance matrix if and only if  $\mathcal{K}^\dagger(D)$  is positive semidefinite. Furthermore, if  $D \in \mathcal{E}^n$ , then*

$$\text{embdim}(D) = \text{rank } \mathcal{K}^\dagger(D) \leq n - 1.$$

This means that given  $D \in \mathcal{E}^n$ , we can find the Gram matrix  $B = \mathcal{K}^\dagger(D)$  and the full rank factorization  $PP^T = B$ . Then the points,  $p_j$ , given by the rows of  $P$  satisfy  $p_j \in \mathbb{R}^t$ , for  $j = 1, \dots, n$ . Moreover,  $t$  is necessarily less than  $n$ .

### 30.4 The Euclidean Distance Matrix Completion Problem

Following [19], we say that an  $n$ -by- $n$  matrix  $D$  is a *partial Euclidean distance matrix* if every entry of  $D$  is either “specified” or “unspecified”,  $\text{diag}(D) = 0$ , and every fully specified principal submatrix of  $D$  is a EDM. Note that this definition implies that every specified entry of  $D$  is nonnegative. In addition, if every fully specified principal submatrix of  $D$  has embedding dimension less than or equal to  $r$ , then we say that  $D$  is a partial EDM in  $\mathbb{R}^r$ .

Associated with an  $n$ -by- $n$  partial EDM  $D$  is a *weighted undirected graph*  $G = (N, E, \omega)$  with node set  $N := \{1, \dots, n\}$ , edge set

$$E := \{ij : i \neq j, \text{ and } D_{ij} \text{ is specified}\},$$

and edge weights  $\omega \in \mathbb{R}_+^E$  with  $\omega_{ij} = \sqrt{D_{ij}}$ , for all  $ij \in E$ . We say that  $H$  is the 0–1 adjacency matrix of  $G$  if  $H \in \mathcal{S}^n$  with

$$H_{ij} = \begin{cases} 1, & ij \in E \\ 0, & ij \notin E. \end{cases}$$

The *Euclidean distance matrix completion (EDMC) problem* asks to find a completion of a partial Euclidean distance matrix  $D$ ; that is, if  $G = (N, E, \omega)$  is the weighted graph associated with  $D$ , the EDMC problem can be posed as

$$\begin{aligned} \text{find} \quad & \hat{D} \in \mathcal{E}^n \\ \text{such that} \quad & \hat{D}_{ij} = D_{ij}, \forall ij \in E. \end{aligned} \tag{30.5}$$

Letting  $H \in \mathcal{S}^n$  be the 0–1 adjacency matrix of  $G$ , the EDMC problem can be stated as

$$\begin{aligned} \text{find} \quad & \hat{D} \in \mathcal{E}^n \\ \text{such that} \quad & H \circ \hat{D} = H \circ D, \end{aligned} \tag{30.6}$$

where “ $\circ$ ” represents the component-wise (or *Hadamard*) matrix product.

Using the linear map  $\mathcal{K}$ , we can substitute  $\hat{D} = \mathcal{K}(Y)$ , where  $Y \in \mathcal{S}_+^n \cap \mathcal{S}_C^n$ , in the EDMC problem (30.6) to obtain the equivalent problem

$$\begin{aligned} \text{find} \quad & Y \in \mathcal{S}_+^n \cap \mathcal{S}_C^n \\ \text{such that} \quad & H \circ \mathcal{K}(Y) = H \circ D. \end{aligned} \tag{30.7}$$

### 30.4.1 The Low-Dimensional EDM Completion Problem

If  $D$  is a partial EDM in  $\mathbb{R}^r$ , one is often interested in finding a Euclidean distance matrix completion of  $D$  that has embedding dimension  $r$ . The *low-dimensional Euclidean distance matrix completion problem* is

$$\begin{aligned} \text{find } \quad & \hat{D} \in \mathcal{E}^n \\ \text{such that } & H \circ \hat{D} = H \circ D \\ \text{embdim}(\hat{D}) &= r, \end{aligned} \tag{30.8}$$

where  $H$  is the 0–1 adjacency matrix of the graph  $G$  associated with  $D$ .

Using the linear map  $\mathcal{K}$ , we can state the low-dimensional EDMC problem (30.8) as the following rank constrained SDP:

$$\begin{aligned} \text{find } \quad & Y \in \mathcal{S}_+^n \\ \text{such that } & H \circ \mathcal{K}(Y) = H \circ D \\ & Ye = 0 \\ & \text{rank}(Y) = r. \end{aligned} \tag{30.9}$$

Note that the constraint  $Ye = 0$  means that there is no positive definite feasible solution. This means that standard interior point methods cannot properly handle this problem without some modification, e.g., the use of  $K_V$  given above in (30.3).

### 30.4.2 Chordal EDM Completions

Let  $G$  be a graph and  $C$  be a cycle in the graph. We say that  $C$  has a *chord* if there are two vertices on  $C$  that are connected by an edge which is not contained in  $C$ . Note that it is necessary that a cycle with a chord have length more than three. The graph  $G$  is called *chordal* if every cycle of the graph with length three or more has a *chord*. In the landmark paper [67], they show the strong result that any partial semidefinite matrix with a chordal graph has a semidefinite completion; moreover, if a graph is not chordal, then there exists a partial semidefinite matrix with that graph, but having no semidefinite completion.

Due to the strong connection between EDMs and semidefinite matrices, it is not surprising that the result of chordal semidefinite completions of [67] extends to the case of chordal EDMC. Indeed, see [19]:

1. Any partial Euclidean distance matrix in  $\mathbb{R}^r$  with a chordal graph can be completed to a distance matrix in  $\mathbb{R}^r$ .
2. Every nonchordal graph has a partial Euclidean distance matrix that does not admit any distance matrix completions.

3. If the graph  $G$  of a partial Euclidean distance matrix  $D$  in  $\mathbb{R}^r$  is chordal, then the completion of  $D$  is unique if and only if

$$\text{rank}\left(\begin{bmatrix} 0 & e^T \\ e & D[S] \end{bmatrix}\right) = r+2, \quad \text{for all minimal vertex separators } S \text{ of } G.$$

A set of vertices  $S$  in a graph  $G$  is a *minimal vertex separator* if removing the vertices  $S$  from  $G$  separates some vertices  $u$  and  $v$  in  $G$ , and no proper subset of  $S$  separates  $u$  and  $v$ . It is discussed in, for example, [84] how the *maximum cardinality search (MCS)* can be used to test in linear time if a graph  $G$  is chordal; moreover, [84] show that MCS can also be used to compute all minimal vertex separators of a chordal graph in linear time. See also [8, 9], for example, for more on the topic of uniqueness of EDM completions.

### 30.4.3 Corresponding Graph Realization Problems

Suppose that we are given an  $n \times n$  partial EDM  $\bar{D}$ , where only the elements  $\bar{D}_{ij}$ , for all  $ij \in E$ , are known. In addition, suppose every fully specified principal submatrix of  $\bar{D}$  has an embedding dimension less or equal to  $r$ . We let  $\mathcal{G} = (N, E, \omega)$  be the corresponding simple weighted graph on the node set  $N = \{1, \dots, n\}$  whose edge set  $E$  corresponds to the known entries of  $\bar{D}$ , with edge weights  $\bar{D}_{ij} = \omega_{ij}^2$ , for all  $ij \in E$ . The *graph realization* problem consists of finding a mapping  $p: N \rightarrow \mathbb{R}^r$ , with  $p_i \in \mathbb{R}^r$  for all  $i \in N$ , such that  $\|p_i - p_j\| = \omega_{ij}$ , for all  $ij \in E$ .

We note that a clique  $C \subseteq N$  of the graph  $\mathcal{G}$  defines a complete subgraph of  $\mathcal{G}$  and this corresponds to a known principal submatrix of  $\bar{D}$ . Cliques play a significant role in a facial reduction algorithm for the SNL problem that we describe in Sect. 30.4.8 below.

We note here the deep connection between the Euclidean distance matrix completion problem and the problem of *graph realization*. Let  $N := \{1, \dots, n\}$ . Given a graph  $G = (N, E, \omega)$  with edge weights  $\omega \in \mathbb{R}_+^E$ , the graph realization problem asks to find a mapping  $p: N \rightarrow \mathbb{R}^r$  such that

$$\|p_i - p_j\| = \omega_{ij}, \quad \text{for all } ij \in E;$$

in this case, we say that  $G$  has an *r-realization*,  $p$ . Clearly the graph realization problem is equivalent to the problem of Euclidean distance matrix completion, and the problem of the *r-realizability* of a weighted graph is equivalent to the low-dimensional Euclidean distance matrix completion problem.

A related problem is that of graph rigidity. Again, let  $N := \{1, \dots, n\}$ . An unweighted graph  $G = (N, E)$  together with a mapping  $p: N \rightarrow \mathbb{R}^r$  is called a *framework* (also *bar framework*) in  $\mathbb{R}^r$ , and is denoted by  $(G, p)$ . Frameworks  $(G, p)$  in  $\mathbb{R}^r$  and  $(G, q)$  in  $\mathbb{R}^s$  are called *equivalent* if

$$\|p_i - p_j\| = \|q_i - q_j\|, \quad \text{for all } ij \in E.$$

Furthermore,  $p$  and  $q$  are called *congruent* if

$$\|p_i - p_j\| = \|q_i - q_j\|, \quad \text{for all } i, j = 1, \dots, n. \quad (30.10)$$

Note that condition (30.10) can be stated as

$$\mathcal{K}(PP^T) = \mathcal{K}(QQ^T),$$

where  $P \in \mathbb{R}^{n \times r}$  and  $Q \in \mathbb{R}^{n \times s}$  are defined as

$$P := \begin{bmatrix} p_1^T \\ \vdots \\ p_n^T \end{bmatrix} \quad \text{and} \quad Q := \begin{bmatrix} q_1^T \\ \vdots \\ q_n^T \end{bmatrix}.$$

Thus, if  $P^T e = Q^T e = 0$ , then we have  $PP^T = QQ^T$ , which implies that:

- $P\bar{Q} = [Q \ 0]$ , for some orthogonal  $\bar{Q} \in \mathbb{R}^r$ , if  $r \geq s$ ;
- $[P \ 0]\bar{Q} = Q$ , for some orthogonal  $\bar{Q} \in \mathbb{R}^s$ , if  $s \geq r$ .

A framework  $(G, p)$  in  $\mathbb{R}^r$  is called *globally rigid* in  $\mathbb{R}^r$  if all equivalent frameworks  $(G, q)$  in  $\mathbb{R}^r$  satisfy condition (30.10). Similarly, a framework  $(G, p)$  in  $\mathbb{R}^r$  is called *universally rigid* in  $\mathbb{R}^r$  if, for all  $s = 1, \dots, n-1$ , all equivalent frameworks  $(G, q)$  in  $\mathbb{R}^s$  satisfy condition (30.10). Note that a framework  $(G, p)$  in  $\mathbb{R}^r$  corresponds to the pair  $(H, D)$ , where  $H$  is the 0–1 adjacency matrix of  $G$ , and  $D$  is a Euclidean distance matrix  $\text{embdim}(D) \leq r$ . Therefore, the framework  $(G, p)$  given by  $(H, D)$  is globally rigid if

$$H \circ \hat{D} = H \circ D \quad \Rightarrow \quad \hat{D} = D,$$

for all  $\hat{D} \in \mathcal{E}^n$  with  $\text{embdim}(\hat{D}) \leq r$ ; equivalently,  $(G, p)$  is globally rigid if

$$H \circ \mathcal{K}(Y) = H \circ D \quad \Rightarrow \quad \mathcal{K}(Y) = D,$$

for all  $Y \in \mathcal{S}_+^n \cap \mathcal{S}_C^n$  with  $\text{rank}(Y) \leq r$ . Moreover, the framework  $(G, p)$  given by  $(H, D)$  is universally rigid if

$$H \circ \hat{D} = H \circ D \quad \Rightarrow \quad \hat{D} = D,$$

for all  $\hat{D} \in \mathcal{E}^n$ ; equivalently,  $(G, p)$  is universally rigid if

$$H \circ \mathcal{K}(Y) = H \circ D \quad \Rightarrow \quad \mathcal{K}(Y) = D,$$

for all  $Y \in \mathcal{S}_+^n \cap \mathcal{S}_C^n$ .

Graph realization and graph rigidity is a vast area of research, so we keep our discussion brief. More information can be found in, for example, [39, 73, 106], the recent survey [3], and the references therein.

### 30.4.4 Low-Dimensional EDM Completion is NP-Hard

We now discuss the complexity of the low-dimensional Euclidean distance matrix completion decision problem (30.8) (equivalently, problem (30.9)). It was independently discovered by [102] and [118] that the problem of graph embeddability with integer edge weights, in  $r = 1$  or  $r = 2$  dimensions, is NP-complete by reduction from the NP-complete problem PARTITION. The PARTITION problem is defined as follows: Given a set of  $S$  of  $n$  integers, determine if there is a partition of  $S$  into two sets  $S_1, S_2 \subseteq S$  such that the sum of the integers in  $S_1$  equals the sum of the integers in  $S_2$ . PARTITION is one of Karp's 21 NP-complete problems in his landmark paper [79].

**Theorem 30.2 ([102, Theorem 3.2]).** *The problem of 1-embeddability of graphs with integer weights is NP-complete.*

Furthermore, by showing that for any  $r$ ,  $r$ -embeddability of  $\{1,2\}$ -weighted graphs is NP-hard, [102] proves that the problem of  $r$ -embeddability of integer weighted graphs is strongly NP-hard.

In practice, the so-called *unit disk graphs* are used; these graphs have realization in some Euclidean space satisfying their edge-weights such that the distance between vertices that are not connected is greater than some *radio range*  $R$ , and  $R$  is greater than all the edge weights. Thus, vertices are connected if and only if they are within radio range. However, see [18], the realizability of unit disk graphs is, in fact, NP-hard (again, by reduction from PARTITION). In the [18] proof, they again work with cycles, which are typically not uniquely realizable. In applications, one is often most interested in unit disk graphs that have a unique realization. However, it is shown in [16, 17] that there is no efficient algorithm for solving the unit disk graph localization problem, even if that graph has a unique realization, unless RP = NP (RP is the class of randomized polynomial time solvable problems). Problems in RP can be solved in polynomial time with high probability using a randomized algorithm. See also [87], for example, for more on the topic of the complexity of Euclidean distance matrix completion and related problems.

Due to these hardness results for the low-dimensional Euclidean distance matrix problem, we turn to convex relaxations which can be solved efficiently, but may not solve our original problem.

### 30.4.5 SDP Relaxation of the Low-Dimensional EDM Completion Problem

The semidefinite relaxation of the low-dimensional EDM completion problem (30.9) is given by relaxing the hard  $\text{rank}(Y) = r$  constraint. Thus, we have the following tractable convex relaxation

$$\begin{aligned}
& \text{find} && Y \in \mathcal{S}_+^n \\
& \text{such that} && H \circ \mathcal{K}(Y) = H \circ D \\
& && Ye = 0.
\end{aligned} \tag{30.11}$$

This semidefinite relaxation essentially allows the points to move into  $\mathbb{R}^k$ , where  $k > r$ . That is, if a solution  $Y$  of problem (30.11) has  $\text{rank}(Y) = k > r$ , then we have found a Euclidean distance matrix completion of  $D$  with embedding dimension  $k$ ; this is even possible if  $D$  has a completion with embedding dimension  $r$ , or even if  $D$  has a *unique* completion with embedding dimension  $r$ .

We can view this relaxation as a Lagrangian relaxation of the low-dimensional EDM completion problem.

**Proposition 30.1** ([82, Proposition 2.48]). *Relaxation (30.11) is the Lagrangian relaxation of Problem (30.9).*

#### 30.4.5.1 Duality of the SDP Relaxation

Problem (30.11) is equivalent to

$$\begin{aligned}
& \text{minimize} && 0 \\
& \text{subject to} && H \circ \mathcal{K}(Y) = H \circ D \\
& && Ye = 0 \\
& && Y \in \mathcal{S}_+^n.
\end{aligned}$$

The Lagrangian of this problem is given by

$$\begin{aligned}
L(Y, \Lambda, v) &= \langle \Lambda, H \circ \mathcal{K}(Y) - H \circ D \rangle + \langle v, Ye \rangle \\
&= \langle \mathcal{K}^*(H \circ \Lambda), Y \rangle - \langle H \circ \Lambda, D \rangle + \langle ev^T, Y \rangle \\
&= \left\langle \mathcal{K}^*(H \circ \Lambda) + \frac{1}{2} (ev^T + ve^T), Y \right\rangle - \langle H \circ \Lambda, D \rangle \\
&= \left\langle \mathcal{K}^*(H \circ \Lambda) + \frac{1}{2} \mathcal{D}_e(v), Y \right\rangle - \langle H \circ \Lambda, D \rangle.
\end{aligned}$$

Therefore, the Lagrangian dual problem is

$$\sup_{\Lambda, v} \inf_{Y \in \mathcal{S}_+^n} L(Y, \Lambda, v),$$

which is equivalent to

$$\sup \left\{ -\langle H \circ \Lambda, D \rangle : \mathcal{K}^*(H \circ \Lambda) + \frac{1}{2} \mathcal{D}_e(v) \succeq 0 \right\}.$$

From this dual problem, we obtain the following partial description of the conjugate face of the minimal face of the EDM completion problem (30.11).

**Proposition 30.2.** *Let  $F := \text{face}(\mathcal{F})$ , where*

$$\mathcal{F} := \{Y \in \mathcal{S}_+^n : H \circ \mathcal{K}(Y) = H \circ D, Ye = 0\}.$$

If  $\mathcal{F} \neq \emptyset$ , then

$$\text{face} \left\{ S \in \mathcal{S}_+^n : S = \mathcal{K}^*(H \circ \Lambda) + \frac{1}{2} \mathcal{D}_e(v), \langle H \circ \Lambda, D \rangle = 0 \right\} \subseteq F^c.$$

Using Proposition 30.2, we obtain the following partial description of the minimal face of the EDM completion problem (30.11).

**Corollary 30.1.** *If  $\mathcal{F} := \{Y \in \mathcal{S}_+^n : H \circ \mathcal{K}(Y) = H \circ D, Ye = 0\} \neq \emptyset$  and there exists  $S \geq 0$  such that*

$$S = \mathcal{K}^*(H \circ \Lambda) + \frac{1}{2} \mathcal{D}_e(v) \quad \text{and} \quad \langle H \circ \Lambda, D \rangle = 0,$$

for some  $\Lambda \in \mathcal{S}^n$  and  $v \in \mathbb{R}^n$ , then

$$\text{face}(\mathcal{F}) \subseteq \mathcal{S}_+^n \cap \{S\}^\perp.$$

For example, we can apply Corollary 30.1 as follows. Let  $\Lambda := 0$  and  $v := e$ . Then,

$$\mathcal{K}^*(H \circ \Lambda) + \frac{1}{2} \mathcal{D}_e(v) = ee^T \geq 0, \quad \text{and} \quad \langle H \circ \Lambda, D \rangle = 0.$$

Thus,

$$\text{face} \{Y \in \mathcal{S}_+^n : H \circ \mathcal{K}(Y) = H \circ D, Ye = 0\} \subseteq \mathcal{S}_+^n \cap \{ee^T\}^\perp = V\mathcal{S}_+^{n-1}V^T,$$

where  $\left[ V \frac{1}{\sqrt{n}} e \right] \in \mathbb{R}^{n \times n}$  is orthogonal. Note that we have  $V^T V = I$  and  $VV^T = J$ , where  $J$  is the orthogonal projector onto  $\{e\}^\perp$ . Therefore, Problem (30.11) is equivalent to the reduced problem

$$\begin{aligned} & \text{find} && Z \in \mathcal{S}_+^{n-1} \\ & \text{such that} && H \circ \mathcal{K}_V(Z) = H \circ D, \end{aligned}$$

where  $\mathcal{K}_V: \mathcal{S}^{n-1} \rightarrow \mathcal{S}^n$  is defined as in (30.3).

### 30.4.6 Rank Minimization Heuristics for the EDM Completion Problem

In order to encourage having a solution of the semidefinite relaxation with low rank, the following heuristic has been suggested by [114] and used with great success

by [26] on the sensor network localization problem. The idea is that we can try to “flatten” the graph associated with a partial Euclidean distance matrix by pushing the nodes of the graph away from each other as much as possible. This flattening of the graph then corresponds to reducing the rank of the semidefinite solution of the relaxation. Geometrically, this makes a lot of sense, and [114] gives this nice analogy: a loose string on the table can occupy two dimensions, but the same string pulled taut occupies just one dimension.

Therefore, we would like to maximize the objective function

$$\sum_{i,j=1}^n \|p_i - p_j\|^2 = e^T \mathcal{K}(PP^T) e,$$

where

$$P := \begin{bmatrix} p_1^T \\ \vdots \\ p_n^T \end{bmatrix},$$

subject to the distance constraints holding. Moreover, if we include the constraint that  $P^T e = 0$ , then

$$\begin{aligned} e^T \mathcal{K}(PP^T) e &= \langle ee^T, \mathcal{K}(PP^T) \rangle \\ &= \langle \mathcal{K}^*(ee^T), PP^T \rangle \\ &= \langle 2(\text{Diag}(ee^T e) - ee^T), PP^T \rangle \\ &= \langle 2(nI - ee^T), PP^T \rangle \\ &= \langle 2nI, PP^T \rangle - \langle ee^T, PP^T \rangle \\ &= 2n \cdot \text{trace}(PP^T). \end{aligned}$$

Note that

$$\text{trace}(PP^T) = \sum_{i=1}^n \|p_i\|^2,$$

so pushing the nodes away from each other is equivalent to pushing the nodes away from the origin, under the assumption that the points are centred at the origin. Normalizing this objective function by dividing by the constant  $2n$ , substituting  $Y = PP^T$ , and relaxing the rank constraint on  $Y$ , we obtain the following *regularized* semidefinite relaxation of the low-dimensional Euclidean distance matrix completion problem:

$$\begin{aligned} &\text{maximize} && \text{trace}(Y) \\ &\text{subject to} && H \circ \mathcal{K}(Y) = H \circ D \\ &&& Y \in \mathcal{S}_+^n \cap \mathcal{S}_C^n. \end{aligned} \tag{30.12}$$

It is very interesting to compare this heuristic with the *nuclear norm* rank minimization heuristic that has received much attention lately. This nuclear norm heuristic has had much success and obtained many practical results for computing minimum-rank solutions of linear matrix equations (for example, finding the exact completion of low-rank matrices); see, for example, [14, 34, 35, 55, 99] and the recent review paper [98].

The *nuclear norm* of a matrix  $X \in \mathbb{R}^{m \times n}$  is given by

$$\|X\|_* := \sum_{i=1}^k \sigma_i(X),$$

where  $\sigma_i(X)$  is the  $i^{\text{th}}$  largest singular value of  $X$ , and  $\text{rank}(X) = k$ . The nuclear norm of a symmetric matrix  $Y \in \mathcal{S}^n$  is then given by

$$\|Y\|_* = \sum_{i=1}^n |\lambda_i(Y)|.$$

Furthermore, for  $Y \in \mathcal{S}_+^n$ , we have  $\|Y\|_* = \text{trace}(Y)$ .

Since we are interested in solving the *rank minimization* problem,

$$\begin{aligned} & \text{minimize} && \text{rank}(Y) \\ & \text{subject to} && H \circ \mathcal{K}(Y) = H \circ D \\ & && Y \in \mathcal{S}_+^n \cap \mathcal{S}_C^n, \end{aligned} \tag{30.13}$$

the nuclear norm heuristic gives us the problem

$$\begin{aligned} & \text{minimize} && \text{trace}(Y) \\ & \text{subject to} && H \circ \mathcal{K}(Y) = H \circ D \\ & && Y \in \mathcal{S}_+^n \cap \mathcal{S}_C^n. \end{aligned} \tag{30.14}$$

However, this is geometrically interpreted as trying to bring all the nodes of the graph as close to the origin as possible. Intuition tells us that this approach would produce a solution with a high embedding dimension.

Another rank minimization heuristic that has been considered recently in [56] is the log-det maximization heuristic. There they successfully computed solutions to Euclidean distance matrix problems with very low embedding dimension via this heuristic.

### 30.4.7 Nearest EDM Problem

For many applications, we are given an approximate (or partial) EDM  $\bar{D}$  and we need to find the *nearest EDM*. Some of the elements of  $\bar{D}$  may be inexact. Therefore, we can model this problem as the norm minimization problem,

$$\begin{aligned} & \text{minimize } \|W \circ (K(B) - \bar{D})\| \\ & \text{subject to } K(B)_{ij} = \bar{D}_{ij}, \forall ij \in E, \\ & B \succeq 0, \end{aligned} \tag{30.15}$$

where  $W$  represents a weight matrix to reflect the accuracy of the data, and  $E$  is a subset of the pairs of nodes corresponding to exact data. This model often includes upper and lower bounds on the distances; see, e.g., [4].

We have not specified the norm in (30.15). The Frobenius norm was used in [4], where small problems were solved; see also [5]. The Frobenius norm was also used in [45, 46], where the EDM was specialized to the sensor network localization, SNL, model. All the above approaches had a difficult time solving large problems. The difficulty was both in the size of the problem and the accuracy of the solutions. It was observed in [46] that the Jacobian of the optimality conditions had many zero singular values at optimality. An explanation of this *degeneracy* is discussed below in Sect. 30.4.8.

### 30.4.8 Facial Reduction

As mentioned in Sect. 30.4.7 above, solving large scale nearest EDM problems using SDP is difficult due to the size of the resulting SDP and also due to the difficulty in getting accurate solutions. In particular, the empirical tests in [46] led to the observation [82, 83] that the problems are highly, implicitly degenerate. In particular, if we have a clique in the data,  $\alpha \subseteq N$ , (equivalently, we have a known principal submatrix of the data  $\bar{D}$ ), then we have the following basic result.

**Theorem 30.3 ([83, Theorem 2.3]).** *Let  $D \in \mathcal{E}^n$ , with embedding dimension  $r$ . Let  $\bar{D} := D[1:k] \in \mathcal{E}^k$  with embedding dimension  $t$ , and  $B := \mathcal{K}^\dagger(\bar{D}) = \bar{U}_B S \bar{U}_B^T$ , where  $\bar{U}_B \in \mathcal{M}^{k \times t}$ ,  $\bar{U}_B^T \bar{U}_B = I_t$ , and  $S \in \mathcal{S}'_{++}$ . Furthermore, let  $U_B := \left[ \bar{U}_B \ \frac{1}{\sqrt{k}} e \right] \in \mathcal{M}^{k \times (t+1)}$ ,  $U := \begin{bmatrix} U_B & 0 \\ 0 & I_{n-k} \end{bmatrix}$ , and let  $V \left[ \begin{smallmatrix} U^T e \\ \|U^T e\| \end{smallmatrix} \right] \in \mathcal{M}^{n-k+t+1}$  be orthogonal. Then*

$$\text{face} \mathcal{K}^\dagger(\mathcal{E}^n(1:k, \bar{D})) = (U \mathcal{S}'_{+}^{n-k+t+1} U^T) \cap \mathcal{S}_C = (UV) \mathcal{S}'_{+}^{n-k+t} (UV)^T.$$

Theorem 30.3 implies that the Slater constraint qualification (strict feasibility) fails if the known set of distances contains a clique. Moreover, we explicitly find

the expression for the face of feasible semidefinite Gram matrices. This means we have reduced the size of the problem from matrices in  $\mathcal{S}^n$  to matrices in  $\mathcal{S}^{n-k+t}$ . We can continue to do this for all disjoint cliques. Note the equivalences between the cliques, the faces, and the subspace representation for the faces.

The following result shows that we can continue to reduce the size of the problem using two intersecting cliques. All we have to do is find the corresponding subspace representations and calculate the intersection of these subspaces.

**Theorem 30.4 ([83, Theorem 2.7]).** *Let  $D \in \mathcal{E}^n$  with embedding dimension  $r$  and, define the sets of positive integers*

$$\begin{aligned}\alpha_1 &:= 1:(\bar{k}_1 + \bar{k}_2), \quad \alpha_2 := (\bar{k}_1 + 1):(\bar{k}_1 + \bar{k}_2 + \bar{k}_3) \subseteq 1:n, \\ k_1 &:= |\alpha_1| = \bar{k}_1 + \bar{k}_2, \quad k_2 := |\alpha_2| = \bar{k}_2 + \bar{k}_3, \\ k &:= \bar{k}_1 + \bar{k}_2 + \bar{k}_3.\end{aligned}$$

For  $i = 1, 2$ , let  $\bar{D}_i := D[\alpha_i] \in \mathcal{E}^{k_i}$  with embedding dimension  $t_i$ , and  $B_i := \mathcal{K}^\dagger(\bar{D}_i) = \bar{U}_i S_i \bar{U}_i^T$ , where  $\bar{U}_i \in \mathcal{M}^{k_i \times t_i}$ ,  $\bar{U}_i^T \bar{U}_i = I_{t_i}$ ,  $S_i \in \mathcal{S}_{++}^{t_i}$ , and  $U_i := \left[ \bar{U}_i \ \frac{1}{\sqrt{k_i}} e \right] \in \mathcal{M}^{k_i \times (t_i+1)}$ . Let  $t$  and  $\bar{U} \in \mathcal{M}^{k \times (t+1)}$  satisfy

$$\mathcal{R}(\bar{U}) = \mathcal{R}\left(\begin{bmatrix} U_1 & 0 \\ 0 & I_{\bar{k}_3} \end{bmatrix}\right) \cap \mathcal{R}\left(\begin{bmatrix} I_{\bar{k}_1} & 0 \\ 0 & U_2 \end{bmatrix}\right), \text{ with } \bar{U}^T \bar{U} = I_{t+1}.$$

Let  $U := \begin{bmatrix} \bar{U} & 0 \\ 0 & I_{n-k} \end{bmatrix} \in \mathcal{M}^{n \times (n-k+t+1)}$  and  $V \frac{U^T e}{\|U^T e\|} \in \mathcal{M}^{n-k+t+1}$  be orthogonal. Then

$$\bigcap_{i=1}^2 \text{face } \mathcal{K}^\dagger(\mathcal{E}^n(\alpha_i, \bar{D}_i)) = (U \mathcal{S}_+^{n-k+t+1} U^T) \cap \mathcal{S}_C = (UV) \mathcal{S}_+^{n-k+t} (UV)^T.$$

Moreover, the intersections of subspaces can be found efficiently and accurately.

**Lemma 30.1 ([83, Lemma 2.9]).** *Let*

$$U_1 := \begin{smallmatrix} r+1 \\ s_1 \\ k \end{smallmatrix} \begin{bmatrix} U'_1 \\ U''_1 \end{bmatrix}, \quad U_2 := \begin{smallmatrix} r+1 \\ s_2 \\ k \end{smallmatrix} \begin{bmatrix} U''_2 \\ U'_2 \end{bmatrix}, \quad \hat{U}_1 := \begin{smallmatrix} r+1 & s_2 \\ s_1 & \\ s_2 & \end{smallmatrix} \begin{bmatrix} U'_1 & 0 \\ U''_1 & 0 \\ 0 & I \end{bmatrix}, \quad \hat{U}_2 := \begin{smallmatrix} r+1 \\ s_1 & r+1 \\ s_2 & \\ s_2 & \end{smallmatrix} \begin{bmatrix} I & 0 \\ 0 & U''_2 \\ 0 & U'_2 \end{bmatrix}$$

be appropriately blocked with  $U''_1, U''_2 \in \mathcal{M}^{k \times (r+1)}$  full column rank and  $\mathcal{R}(U''_1) = \mathcal{R}(U''_2)$ . Furthermore, let

$$\bar{U}_1 := \begin{smallmatrix} r+1 \\ s_1 \\ s_2 \end{smallmatrix} \begin{bmatrix} U'_1 \\ U''_1 \\ U'_2(U''_2)^\dagger U''_1 \end{bmatrix}, \quad \bar{U}_2 := \begin{smallmatrix} r+1 \\ s_1 \\ s_2 \end{smallmatrix} \begin{bmatrix} U'_1(U''_1)^\dagger U''_2 \\ U''_2 \\ U'_2 \end{bmatrix}.$$

Then  $\bar{U}_1$  and  $\bar{U}_2$  are full column rank and satisfy

$$\mathcal{R}(\hat{U}_1) \cap \mathcal{R}(\hat{U}_2) = \mathcal{R}(\bar{U}_1) = \mathcal{R}(\bar{U}_2).$$

Moreover, if  $e_{r+1} \in \mathbb{R}^{r+1}$  is the  $(r+1)^{\text{st}}$  standard unit vector, and  $U_i e_{r+1} = \alpha_i e$ , for some  $\alpha_i \neq 0$ , for  $i = 1, 2$ , then  $\bar{U}_i e_{r+1} = \alpha_i e$ , for  $i = 1, 2$ .

As mentioned above, a clique  $\alpha$  corresponds to both a principal submatrix  $\bar{D}[\alpha]$ , in the data  $\bar{D}$ ; to a face in  $\mathcal{S}_+^n$ , and to a subspace. In addition, we can use  $\mathcal{K}$  to obtain  $B = \mathcal{K}^\dagger(\bar{D}[\alpha]) \succeq 0$  to represent the face and then use the factorization  $B = PP^T$  to find a point representation. Using this point representation increases the accuracy of the calculations.

### 30.4.9 Minimum Norm Feasibility Problem

The (best) least squares feasible solution was considered in [2]. We can replace  $\mathcal{K}$  with  $\mathcal{L}$  in (30.4). We get

$$\begin{aligned} & \text{minimize} && \|X\|_F^2 \\ & \text{subject to} && \mathcal{L}(X)_{ij} = \bar{D}_{ij}, \quad \forall ij \in E \\ & && X \geq 0. \end{aligned} \tag{30.16}$$

The Lagrangian dual is particularly elegant and can be solved efficiently. In many cases, these (large, sparse case) problems can essentially be solved explicitly.

## 30.5 Applications

There are many applications of Euclidean distance matrices, including wireless sensor network localization, molecular conformation in chemistry and bioinformatics, and nonlinear dimensionality reduction in statistics and machine learning. For space consideration, we emphasize sensor network localization. And, in particular, we look at methods that are based on the EDM problem and compare these with the Biswas–Ye SDP relaxation.

### 30.5.1 Sensor Network Localization

The *sensor network localization (SNL)* problem is a low-dimensional Euclidean distance matrix completion problem in which the position of a subset of the nodes is specified. Typically, a wireless ad hoc sensor network consists of  $n$  sensors in e.g., a geographical area. Each sensor has wireless communication capability and

the ability for some signal processing and networking. Applications abound, for example: military; detection and characterization of chemical, biological, radiological, nuclear, and explosive attacks; monitoring environmental changes in plains, forests, oceans, etc.; monitoring vehicle traffic; providing security in public facilities; etc...

We let  $x_1, \dots, x_{n-m} \in \mathbb{R}^r$  denote the unknown *sensor* locations; while  $a_1 = x_{n-m+1}, \dots, a_m = x_n \in \mathbb{R}^r$  denotes the known positions of the *anchors/beacons*. Define:

$$X := \begin{bmatrix} x_1^T \\ \vdots \\ x_{n-m}^T \end{bmatrix} \in \mathbb{R}^{(n-m) \times r}; \quad A := \begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix} \in \mathbb{R}^{m \times r}; \quad P := \begin{bmatrix} X \\ A \end{bmatrix} \in \mathbb{R}^{n \times r}.$$

We partition the  $n$ -by- $n$  partial Euclidean distance matrix as

$$D =: \begin{smallmatrix} n-m & m \\ \hline m & m \end{smallmatrix} \begin{bmatrix} D_{11} & D_{12} \\ D_{12}^T & D_{22} \end{bmatrix}.$$

The sensor network localization problem can then be stated as follows.

Given:  $A \in \mathbb{R}^{m \times r}$ ;  $D \in \mathcal{S}^n$  a partial Euclidean distance matrix satisfying  $D_{22} = \mathcal{K}(AA^T)$ , with corresponding 0-1 adjacency matrix  $H$ ;

$$\begin{aligned} \text{find} \quad & X \in \mathbb{R}^{(n-m) \times r} \\ \text{such that} \quad & H \circ \mathcal{K}(PP^T) = H \circ D \\ & P = \begin{bmatrix} X \\ A \end{bmatrix} \in \mathbb{R}^{n \times r}. \end{aligned} \tag{30.17}$$

Before discussing the semidefinite relaxation of the sensor network localization problem, we first discuss the importance of the anchors, and how we may, in fact, ignore the constraint on  $P$  that its bottom block must equal the anchor positions  $A$ .

### 30.5.1.1 Anchors and the Procrustes Problem

For the uniqueness of the sensor positions, a key assumption that we need to make is that the affine hull of the anchors  $A \in \mathbb{R}^{m \times r}$  is full-dimensional,  $\text{aff}\{a_1, \dots, a_m\} = \mathbb{R}^r$ . This further implies that  $m \geq r + 1$ , and  $A$  is full column rank.

On the other hand, from the following observations, we see that we can ignore the constraint  $P_2 = A$  in problem (30.17), where  $P \in \mathbb{R}^{n \times r}$  is partitioned as

$$P =: \begin{smallmatrix} n-m & r \\ \hline m & m \end{smallmatrix} \begin{bmatrix} P_1 \\ P_2 \end{bmatrix}. \tag{30.18}$$

The distance information in  $D_{22}$  is sufficient for completing the partial EDM; i.e., if  $P$  satisfies  $H \circ \mathcal{K}(PP^T) = H \circ D$  in problem (30.17), then

$$\mathcal{K}(P_2 P_2^T) = \mathcal{K}(AA^T).$$

Assuming, without loss of generality, that  $P_2^T e = 0$  and  $A^T e = 0$ , we have that  $P_2 P_2^T = AA^T$ . As we now see, this implies that there exists an orthogonal  $Q \in \mathbb{R}^r$  such that  $P_2 Q = A$ . Since such an orthogonal transformation does not change the distances between points, we have that  $PQ$  is a feasible solution of the sensor network localization problem (30.17), with sensor positions  $X := P_1 Q$ , i.e., we can safely ignore the positions of the anchors until after the missing distances have been found.

The existence of such an orthogonal transformation follows from the classical Procrustes problem. Given  $A, B \in \mathbb{R}^{m \times n}$ , solve:

$$\begin{aligned} & \text{minimize} && \|BQ - A\|_F \\ & \text{subject to} && Q^T Q = I. \end{aligned} \quad (30.19)$$

The general solution to this problem was first given in [105] (see also [62, 66, 75]).

**Theorem 30.5 ([105]).** *Let  $A, B \in \mathbb{R}^{m \times n}$ . Then  $Q := UV^T$  is an optimal solution of the Procrustes problem (30.19), where  $B^T A = U \Sigma V^T$  is the singular value decomposition of  $B^T A$ .*

From Theorem 30.5, we have the following useful consequence.

**Proposition 30.3 ([82, Proposition 3.2]).** *Let  $A, B \in \mathbb{R}^{m \times n}$ . Then  $AA^T = BB^T$  if and only if there exists an orthogonal  $Q \in \mathbb{R}^{n \times n}$  such that  $BQ = A$ .*

Therefore, the method of discarding the constraint  $P_2 = A$  discussed above is justified. This suggests a simple approach for solving the sensor network localization problem. First we solve the equivalent low-dimensional Euclidean distance matrix completion problem,

$$\begin{aligned} & \text{find} && \bar{Y} \in \mathcal{S}_+^n \cap \mathcal{S}_C^n \\ & \text{such that} && H \circ \mathcal{K}(\bar{Y}) = H \circ D \\ & && \text{rank}(\bar{Y}) = r. \end{aligned} \quad (30.20)$$

Note that without the anchor constraint, we can now assume that our points are centred at the origin, hence the constraint  $\bar{Y} \in \mathcal{S}_C^n$ . Factoring  $\bar{Y} = PP^T$ , for some  $P \in \mathbb{R}^{n \times r}$ , we then translate  $P$  so that  $P_2^T e = 0$ . Similarly, we translate the anchors so that  $A^T e = 0$ . We then apply an orthogonal transformation to align  $P_2$  with the anchors positions in  $A$  by solving a Procrustes problem using the solution technique in Theorem 30.5. Finally, if we translated to centre the anchors, we simply translate everything back accordingly.

However, as problems with rank constraints are often NP-hard, problem (30.20) may be very hard to solve. In fact, we saw in Sect. 30.4.4 that the general problem (30.20) can be reduced to the NP-complete problem PARTITION. Therefore, we now turn to investigating semidefinite relaxations of the sensor network localization problem.

Based on our discussion in this section, the relaxation that immediately comes to mind is the semidefinite relaxation of the low-dimensional Euclidean distance matrix completion problem (30.9), which is given by relaxing the rank constraint in problem (30.20); that is, we get the relaxation

$$\begin{aligned} \text{find} \quad & \bar{Y} \in \mathcal{S}_+^n \cap \mathcal{S}_C^n \\ \text{such that} \quad & H \circ \mathcal{K}(\bar{Y}) = H \circ D. \end{aligned} \quad (30.21)$$

However, as we will see in the next section, it is possible to take advantage of the structure available in the constraints corresponding to the anchor-anchor distances. We will show how this structure allows us to reduce the size of the semidefinite relaxation.

### 30.5.1.2 Semidefinite Relaxation of the SNL Problem

To get a semidefinite relaxation of the sensor network localization problem, we start by writing problem (30.17) as,

$$\begin{aligned} \text{find} \quad & X \in \mathbb{R}^{(n-m) \times r} \\ \text{such that} \quad & H \circ \mathcal{K}(\bar{Y}) = H \circ D \\ & \bar{Y} = \begin{bmatrix} XX^T & XA^T \\ AX^T & AA^T \end{bmatrix}. \end{aligned} \quad (30.22)$$

Next we show that the nonlinear second constraint on  $\bar{Y}$ , the block-matrix constraint, may be replaced by a semidefinite constraint, a linear constraint, and a rank constraint on  $\bar{Y}$ .

**Proposition 30.4 ([46, 83]).** *Let  $A \in \mathbb{R}^{m \times r}$  have full column rank, and let  $\bar{Y} \in \mathcal{S}^n$  be partitioned as*

$$\bar{Y} = \begin{smallmatrix} n-m & m \\ \hline m & \end{smallmatrix} \begin{bmatrix} \bar{Y}_{11} & \bar{Y}_{12} \\ \bar{Y}_{12}^T & \bar{Y}_{22} \end{bmatrix}.$$

*Then the following hold:*

1. *If  $\bar{Y}_{22} = AA^T$  and  $\bar{Y} \geq 0$ , then there exists  $X \in \mathbb{R}^{(n-m) \times r}$  such that*

$$\bar{Y}_{12} = XA^T,$$

*and  $X$  is given uniquely by  $X = \bar{Y}_{12}A^{\dagger T}$ .*

2. There exists  $X \in \mathbb{R}^{(n-m) \times r}$  such that  $\bar{Y} = \begin{bmatrix} XX^T & XA^T \\ AX^T & AA^T \end{bmatrix}$  if and only if  $\bar{Y}$  satisfies

$$\left\{ \begin{array}{l} \bar{Y} \succeq 0 \\ \bar{Y}_{22} = AA^T \\ \text{rank}(\bar{Y}) = r \end{array} \right\}.$$

Now, by Proposition 30.4, we have that problem (30.17) is equivalent to

$$\begin{aligned} \text{find} \quad & \bar{Y} \in \mathcal{S}_+^n \\ \text{such that} \quad & H \circ \mathcal{K}(\bar{Y}) = H \circ D \\ & \bar{Y}_{22} = AA^T \\ & \text{rank}(\bar{Y}) = r. \end{aligned} \tag{30.23}$$

Relaxing the hard rank constraint, we obtain the semidefinite relaxation of the sensor network localization problem:

$$\begin{aligned} \text{find} \quad & \bar{Y} \in \mathcal{S}_+^n \\ \text{such that} \quad & H \circ \mathcal{K}(\bar{Y}) = H \circ D \\ & \bar{Y}_{22} = AA^T. \end{aligned} \tag{30.24}$$

As in Proposition 30.1, this relaxation is equivalent to the Lagrangian relaxation of the sensor network localization problem (30.23). Moreover, this relaxation essentially allows the sensors to move into a higher dimension. To obtain a solution in  $\mathbb{R}^r$ , we may either project the positions in the higher dimension onto  $\mathbb{R}^r$ , or we can try a best rank- $r$  approximation approach.

### 30.5.1.3 Further Transformations of the SDP Relaxation of the SNL Problem

From Proposition 30.4, we have that  $\bar{Y} \succeq 0$  and  $\bar{Y}_{22} = AA^T$  implies that

$$\bar{Y} = \begin{bmatrix} Y & XA^T \\ AX^T & AA^T \end{bmatrix}$$

for some  $Y \in \mathcal{S}_+^{n-m}$  and  $X = \bar{Y}_{12}A^{\dagger T} \in \mathbb{R}^{(n-m) \times r}$ . Now we make the key observation that having anchors in a Euclidean distance matrix problem implies that our feasible points are restricted to a face of the semidefinite cone. This is because, if  $\bar{Y}$  is feasible for problem (30.24), then

$$\bar{Y} = \begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix} \begin{bmatrix} Y & X \\ X^T & I \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix}^T \in U_A \mathcal{S}_+^{n-m+r} U_A^T, \tag{30.25}$$

where

$$U_A := \begin{smallmatrix} n-m \\ m \end{smallmatrix} \begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix} \in \mathbb{R}^{n \times (n-m+r)}. \quad (30.26)$$

The fact that we must have

$$\begin{smallmatrix} n-m & r \\ n-m & \\ r & \end{smallmatrix} \begin{bmatrix} Y & X \\ X^T & I \end{bmatrix} \in \mathcal{S}_+^{n-m+r}$$

follows from  $\bar{Y} \geq 0$  and the assumption that  $A$  has full column rank (hence  $U_A$  has full column rank). Therefore, we obtain the following reduced problem:

$$\begin{aligned} \text{find } \quad Z &\in \mathcal{S}_+^{n-m+r} \\ \text{such that } \quad H \circ \mathcal{K}(U_A Z U_A^T) &= H \circ D \\ Z_{22} &= I, \end{aligned} \quad (30.27)$$

where  $Z$  is partitioned as

$$Z = \begin{smallmatrix} n-m & r \\ n-m & \\ r & \end{smallmatrix} \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{12}^T & Z_{22} \end{bmatrix}. \quad (30.28)$$

Since  $Y - XX^T$  is the *Schur complement* of the matrix

$$\begin{bmatrix} Y & X \\ X^T & I \end{bmatrix}$$

with respect to the positive definite identity block, we have that

$$Y \geq XX^T \Leftrightarrow \begin{bmatrix} Y & X \\ X^T & I \end{bmatrix} \succeq 0.$$

Therefore, we have a choice of a larger linear semidefinite constraint, or a smaller quadratic semidefinite constraint. See [38] for a theoretical discussion on the barriers associated with these two representations; see [46] for a numerical comparison.

### 30.5.1.4 The Biswas–Ye Formulation

We now present the Biswas–Ye formulation [25–29] of the semidefinite relaxation of the sensor network localization problem. First we let  $Y := XX^T$  be the Gram

matrix of the rows of the matrix  $X \in \mathbb{R}^{(n-m) \times r}$ . Letting  $e_{ij} \in \mathbb{R}^{n-m}$  be the vector with 1 in the  $i$ th position,  $-1$  in the  $j$ th position, and zero elsewhere, we have

$$\begin{aligned}\|x_i - x_j\|^2 &= x_i^T x_i + x_j^T x_j - 2x_i^T x_j \\ &= e_{ij}^T X X^T e_{ij} \\ &= \langle e_{ij} e_{ij}^T, Y \rangle,\end{aligned}$$

for all  $i, j = 1, \dots, n-m$ . Furthermore, letting  $e_i \in \mathbb{R}^{n-m}$  be the vector with 1 in the  $i$ th position, and zero elsewhere. Then

$$\begin{aligned}\|x_i - a_j\|^2 &= x_i^T x_i + a_j^T a_j - 2x_i^T a_j \\ &= \begin{bmatrix} e_i \\ a_j \end{bmatrix}^T \begin{bmatrix} X \\ I \end{bmatrix} \begin{bmatrix} X \\ I \end{bmatrix}^T \begin{bmatrix} e_i \\ a_j \end{bmatrix} \\ &= \left\langle \begin{bmatrix} e_i \\ a_j \end{bmatrix} \begin{bmatrix} e_i \\ a_j \end{bmatrix}^T, \begin{bmatrix} Y & X \\ X^T & I \end{bmatrix} \right\rangle,\end{aligned}$$

for all  $i = 1, \dots, n-m$ , and  $j = n-m+1, \dots, n$ , where we are now considering the anchors  $a_j$  to be indexed by  $j \in \{n-m+1, \dots, n\}$ . Let  $G = (N, E)$  be the graph corresponding to the partial Euclidean distance matrix  $D$ . Let

$$E_x := \{ij \in E : 1 \leq i < j \leq n-m\}$$

be the set of edges between sensors, and let

$$E_a := \{ij \in E : 1 \leq i \leq n-m, n-m+1 \leq j \leq n\}$$

be the set of edges between sensors and anchors. The sensor network localization problem can then be stated as:

$$\begin{aligned}&\text{find} && X \in \mathbb{R}^{(n-m) \times r} \\ &\text{such that} && \begin{aligned} &\left\langle \begin{bmatrix} e_{ij} \\ 0 \end{bmatrix} \begin{bmatrix} e_{ij} \\ 0 \end{bmatrix}^T, Z \right\rangle = D_{ij}, \quad \forall ij \in E_x \\ &\left\langle \begin{bmatrix} e_i \\ a_j \end{bmatrix} \begin{bmatrix} e_i \\ a_j \end{bmatrix}^T, Z \right\rangle = D_{ij}, \quad \forall ij \in E_a \\ &Z = \begin{bmatrix} Y & X \\ X^T & I \end{bmatrix} \\ &Y = XX^T.\end{aligned}\end{aligned} \tag{30.29}$$

The Biswas–Ye semidefinite relaxation of the sensor network localization problem is then formed by relaxing the hard constraint  $Y = XX^T$  to the convex constraint  $Y \succeq XX^T$ . As mentioned above,  $Y \succeq XX^T$  is equivalent to

$$\begin{bmatrix} Y & X \\ X^T & I \end{bmatrix} \succeq 0. \tag{30.30}$$

Therefore, the Biswas–Ye relaxation is the *linear semidefinite optimization problem*

$$\begin{aligned} \text{find} \quad & Z \in \mathcal{S}_+^{n-m+r} \\ \text{such that} \quad & \left\langle \begin{bmatrix} e_{ij} \\ 0 \end{bmatrix} \begin{bmatrix} e_{ij} \\ 0 \end{bmatrix}^T, Z \right\rangle = D_{ij}, \quad \forall ij \in E_x \\ & \left\langle \begin{bmatrix} e_i \\ a_j \end{bmatrix} \begin{bmatrix} e_i \\ a_j \end{bmatrix}^T, Z \right\rangle = D_{ij}, \quad \forall ij \in E_a \\ & Z_{22} = I, \end{aligned} \tag{30.31}$$

where  $Z$  is partitioned as in (30.28). Clearly, we have that the Biswas–Ye formulation (30.31) is equivalent to the Euclidean distance matrix formulation (30.27). This is, in fact, identical to the relaxation in (30.25). The advantage of this point of view is that the approximations of the sensor positions are taken from the matrix  $X$  in (30.30) directly. However, it is not necessarily the case that these approximations are better than using the  $P$  obtained from a best rank- $r$  approximation of  $Y$  in (30.25). (See also the section on p. 906, below.) In fact, the empirical evidence in [46] indicate the opposite is true. Thus, this further emphasizes the fact that the anchors can be ignored.

### 30.5.1.5 Unique Localizability

The sensor network localization problem (30.17)/(30.29) is called *uniquely localizable* if there is a unique solution  $X \in \mathbb{R}^{(n-m) \times r}$  for problem (30.17)/(30.29) and if  $\bar{X} \in \mathbb{R}^{(n-m) \times h}$  is a solution to the problem with anchors  $\bar{A} := [A \ 0] \in \mathbb{R}^{m \times h}$ , then  $\bar{X} = [X \ 0]$ . The following theorem from [108] shows that the semidefinite relaxation is tight if and only if the sensor network localization problem is uniquely localizable.

**Theorem 30.6 ([108, Theorem 2]).** *Let  $A \in \mathbb{R}^{m \times r}$  such that  $[A \ e]$  has full column rank. Let  $D$  be an  $n$ -by- $n$  partial Euclidean distance matrix satisfying  $D_{22} = \mathcal{K}(AA^T)$ , with corresponding graph  $G$  and 0–1 adjacency matrix  $H$ . If  $G$  is connected, the following are equivalent.*

1. *The sensor network localization problem (30.17)/(30.29) is uniquely localizable.*
2. *The max-rank solution of the relaxation (30.27)/(30.31) has rank  $r$ .*
3. *The solution matrix  $Z$  of the relaxation (30.27)/(30.31) satisfies  $Y = XX^T$ , where*

$$Z = \begin{bmatrix} Y & X \\ X^T & I \end{bmatrix}.$$

Therefore, Theorem 30.6 implies that we can solve *uniquely localizable* instances of the sensor network localization problem in polynomial time by solving the semidefinite relaxation (30.27)/(30.31). However, it is important to point out an instance of the sensor network localization problem (30.17)/(30.29) which has a

unique solution in  $\mathbb{R}^r$  need not be uniquely localizable. This is especially important to point out in light of the complexity result in [17] and [16] in which it is proved that there is no efficient algorithm to solve instances of the sensor network localization problem having a unique solution in  $\mathbb{R}^r$ , unless  $RP = NP$ . This means we have two types of sensor network localization problem instances that have a unique solution in  $\mathbb{R}^r$ : (1) uniquely localizable, having no non-congruent solution in a higher dimension; (2) not uniquely localizable, having a non-congruent solution in a higher dimension. Type (1) instances can be solved in polynomial time. Type (2) cannot be solved in polynomial time, unless  $RP = NP$ .

### 30.5.1.6 Obtaining Sensor Positions from the Semidefinite Relaxation

Often it can be difficult to obtain a low rank solution from the semidefinite relaxation of a combinatorial optimization problem. An example of a successful semidefinite rounding technique is the impressive result in [60] for the Max-CUT problem; however, this is not always possible. For the sensor network localization problem we must obtain sensor positions from a solution of the semidefinite relaxation.

In the case of the Biswas–Ye formulation (30.31), or equivalently the Euclidean distance matrix formulation (30.27), the sensor positions  $X \in \mathbb{R}^{(n-m) \times r}$  are obtained from a solution  $Z \in \mathcal{S}_+^{n-m+r}$  by letting  $X := Z_{12}$ , where  $Z$  is partitioned as in (30.28). By Proposition 30.4, under the constraints that  $\bar{Y} \geq 0$  and  $\bar{Y}_{22} = AA^T$ , we may also compute the sensor positions as  $X := \bar{Y}_{12}A^{\dagger T}$ . Clearly, these are equivalent methods for computing the sensor positions.

Just after discussing the Procrustes problem (30.19), another method for computing the sensor positions was discussed for the uniquely localizable case when  $\text{rank}(\bar{Y}) = r$ . Now suppose that  $\text{rank}(\bar{Y}) > r$ . In this case we find a best rank- $r$  approximation of  $\bar{Y}$ . For this, we turn to the classical result of Eckart and Young [51].

**Theorem 30.7 ([30, Theorem 1.2.3]).** *Let  $A \in \mathbb{R}^{m \times n}$  and  $k := \text{rank}(A)$ . Let*

$$A = U\Sigma V^T = \sum_{i=1}^k \sigma_i u_i v_i^T$$

*be the singular value decomposition of  $A$ . Then the unique optimal solution of*

$$\min \{ \|A - X\|_F : \text{rank}(X) = r\}$$

*is given by*

$$X := \sum_{i=1}^r \sigma_i u_i v_i^T,$$

*with  $\|A - X\|_F^2 = \sum_{i=r+1}^k \sigma_i^2$ .*

Note that  $\bar{Y} \geq 0$ , so the eigenvalue decomposition  $\bar{Y} = UDU^T$  is also the singular value decomposition of  $\bar{Y}$ , and is less expensive to compute.

### 30.5.1.7 Comparing Two Methods

Let  $A \in \mathbb{R}^{m \times r}$  and  $D$  be an  $n$ -by- $n$  partial Euclidean distance matrix with

$$D = \begin{bmatrix} D_{11} & D_{12} \\ D_{12}^T & \mathcal{K}(AA^T) \end{bmatrix}.$$

Let  $H$  be the 0–1 adjacency matrix corresponding to  $D$ . Suppose  $D$  has a completion  $\bar{D} \in \mathcal{E}^n$  having  $\text{embdim}(\bar{D}) = r$ . Suppose  $Z$  is a feasible solution of the semidefinite relaxation (30.31) (or equivalently, the Euclidean distance matrix formulation (30.27)).

Using a path-following interior-point method to find  $Z$  can result in a solution with high rank. Indeed, [61] show that for semidefinite optimization problems having strict complementarity, the central path converges to the analytic centre of the optimal solution set (that is, the optimal solution with maximum determinant).

Let  $\bar{Y} := U_A Z U_A^T$ , where  $U_A$  is as defined in (30.26). Suppose that  $k := \text{rank}(\bar{Y}) > r$ . Then  $\mathcal{K}(\bar{Y})$  is a Euclidean distance matrix completion of the partial Euclidean distance matrix  $D$ , with  $\text{embdim}\mathcal{K}(\bar{Y}) = k$ . Moreover,  $\bar{Y} \in \mathcal{S}_+^n$  and  $\bar{Y}_{22} = AA^T$ , so by Proposition 30.4, we have that

$$\bar{Y} = \begin{bmatrix} Y & XA^T \\ AX^T & AA^T \end{bmatrix},$$

where  $Y := Z_{11} = \bar{Y}_{11}$  and  $X := Z_{12} = \bar{Y}_{12}A^{\dagger T}$ . Let  $\bar{Y}_r \in \mathcal{S}_+^n$  be the nearest rank- $r$  matrix to  $\bar{Y} \in \mathcal{S}_+^n$ , in the sense of Theorem 30.7. Let  $\bar{P} \in \mathbb{R}^{n \times k}$  and  $\bar{P}_r \in \mathbb{R}^{n \times r}$  such that

$$\bar{Y} = \bar{P}\bar{P}^T \quad \text{and} \quad \bar{Y}_r = \bar{P}_r\bar{P}_r^T.$$

Let  $\bar{P}$  and  $\bar{P}_r$  be partitioned as

$$\bar{P} = \begin{smallmatrix} r & k-r \\ \vdots & \vdots \\ \bar{P}_{11} & \bar{P}_{12} \\ \bar{P}_{12}^T & \bar{P}_{22} \end{smallmatrix} \quad \text{and} \quad \bar{P}_r = \begin{smallmatrix} r \\ \vdots \\ \bar{P}'_r \\ \bar{P}''_r \end{smallmatrix}.$$

For the first method, we simply use  $X$  for the sensor positions. Since  $[\bar{P}_{12}^T \bar{P}_{22}]^T = \bar{Y}_{22} = AA^T = [A \ 0][A \ 0]^T$ , there exists an orthogonal matrix  $\bar{Q} \in \mathbb{R}^{k \times k}$  such that  $[\bar{P}_{12}^T \bar{P}_{22}]\bar{Q} = [A \ 0]$ . Let  $\hat{P} := \bar{P}\bar{Q}$  and define the partition

$$\hat{P} = \begin{smallmatrix} r & k-r \\ \vdots & \vdots \\ \hat{P}_{11} & \hat{P}_{12} \\ A & 0 \end{smallmatrix}.$$

Therefore, we see that the semidefinite relaxation of the sensor network localization problem has allowed the sensors to move into the higher dimension of  $\mathbb{R}^k$  instead of fixing the sensors to the space  $\mathbb{R}^r$ . Now we have that

$$\bar{Y} = \hat{P}\hat{P}^T = \begin{bmatrix} \hat{P}_{11}\hat{P}_{11}^T + \hat{P}_{12}\hat{P}_{12}^T & \hat{P}_{11}A^T \\ A\hat{P}_{11}^T & AA^T \end{bmatrix},$$

implying that  $\hat{P}_{11} = X$ , and  $Y = XX^T + \hat{P}_{12}\hat{P}_{12}^T$ , so  $Y - XX^T = \hat{P}_{12}\hat{P}_{12}^T \succeq 0$ . Therefore, we have that

$$\left\| \bar{Y} - \begin{bmatrix} X \\ A \end{bmatrix} \begin{bmatrix} X \\ A \end{bmatrix}^T \right\|_F = \left\| \hat{P}_{12}\hat{P}_{12}^T \right\|_F = \left\| Y - XX^T \right\|_F.$$

For the second method, we use  $\bar{X} := \bar{P}_r''Q_r$  for the sensor positions, where  $Q_r \in \mathbb{R}^{r \times r}$  is an orthogonal matrix that minimizes  $\|\bar{P}_r''Q_r - A\|_F$ . Furthermore, we let  $\bar{A} := \bar{P}_r''Q_r$ . Therefore, we have the following relationship between the two different approaches for computing sensor positions:

$$\left\| \bar{Y} - \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix} \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix}^T \right\|_F = \left\| \bar{Y} - \bar{P}_r\bar{P}_r^T \right\|_F \leq \left\| \bar{Y} - \begin{bmatrix} X \\ A \end{bmatrix} \begin{bmatrix} X \\ A \end{bmatrix}^T \right\|_F.$$

Moreover, computational experiments given at the end of this section show that we typically have

$$\left\| \bar{Y} - \begin{bmatrix} \bar{X} \\ A \end{bmatrix} \begin{bmatrix} \bar{X} \\ A \end{bmatrix}^T \right\|_F \approx \left\| \bar{Y} - \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix} \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix}^T \right\|_F.$$

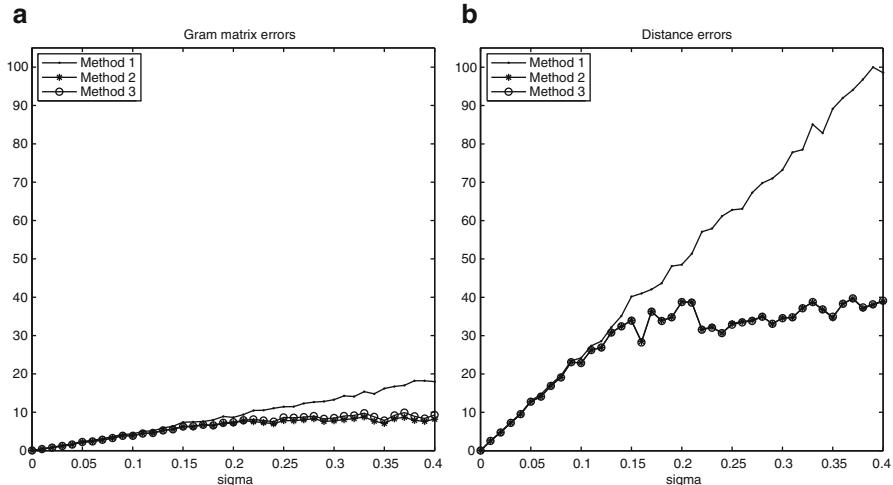
Note that we have no guarantee that  $X$  from the first method, or  $\bar{X}$  from the second method, satisfy the distance constraints. Moreover, we have the following bounds on the approximation of the Euclidean distance matrix  $\mathcal{K}(\bar{Y})$ :

$$\begin{aligned} \left\| \mathcal{K}(\bar{Y}) - \mathcal{K} \left( \begin{bmatrix} X \\ A \end{bmatrix} \begin{bmatrix} X \\ A \end{bmatrix}^T \right) \right\|_F &\leq \|\mathcal{K}\|_F \left\| \bar{Y} - \begin{bmatrix} X \\ A \end{bmatrix} \begin{bmatrix} X \\ A \end{bmatrix}^T \right\|_F \\ &= 2\sqrt{n} \|Y - XX^T\|_F; \end{aligned} \quad (30.32)$$

$$\begin{aligned} \left\| \mathcal{K}(\bar{Y}) - \mathcal{K} \left( \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix} \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix}^T \right) \right\|_F &\leq \|\mathcal{K}\|_F \left\| \bar{Y} - \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix} \begin{bmatrix} \bar{X} \\ \bar{A} \end{bmatrix}^T \right\|_F \\ &= 2\sqrt{n} \left( \sum_{i=r+1}^k \lambda_i^2(\bar{Y}) \right)^{1/2}. \end{aligned} \quad (30.33)$$

Clearly, the upper bound (30.33) for the second method is lower than upper bound (30.32) for the first method, but this need not imply that the second method gives a better  $r$ -dimensional approximation of the  $k$ -dimensional Euclidean distance matrix  $\mathcal{K}(\bar{Y})$ . Indeed, numerical tests were run in [46] to compare these two methods and often found better results using  $\bar{X}$  than when using  $X$ , but this was not always the case.

In Fig. 30.1 we have given the results of a simple numerical test conducted to investigate the differences between Method 1 ( $P = [X; A]$ ), Method 2 ( $P = [\bar{X}; \bar{A}]$ ), and Method 3 ( $P = [\bar{X}; A]$ ). In all three cases, we compared the error



**Fig. 30.1** Method 1 ( $P = [X; A]$ ), Method 2 ( $P = [\bar{X}; \bar{A}]$ ), and Method 3 ( $P = [\bar{X}; A]$ ). The Gram matrix error is  $\|\bar{Y} - PP^T\|_F$  and the distance error is  $\|\mathcal{K}(\bar{Y}) - \mathcal{K}(PP^T)\|_F$ , both normalized so that the maximum error is 100. We use  $n = 100$  and  $m = 10$  (90 sensors and 10 anchors) in dimension  $r = 2$ ; ten of the sensors are inaccurately placed in dimension  $k = 20$ , to various degrees based on  $Y - XX^T = \sigma \hat{P}_{12} \hat{P}_{12}^T$  for a randomly generated matrix  $\hat{P}_{12} \in \mathbb{R}^{90 \times 18}$  with exactly ten nonzero rows

in the approximation of the Gram matrix  $\bar{Y}$  ( $\|\bar{Y} - PP^T\|_F$ ), and the error in the approximation of the Euclidean distance matrix  $\mathcal{K}(\bar{Y})$  ( $\|\mathcal{K}(\bar{Y}) - \mathcal{K}(PP^T)\|_F$ ). In this test, we use 90 sensors and 10 anchors in dimension  $r = 2$ . We had ten of the sensors inaccurately placed in dimension  $k = 20$ , to various degrees. We see that Method 2 and Method 3 are almost identical and may improve the approximation error when some sensors are inaccurately placed.

## References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: a survey. *Comput. Netw.* **38**, 393–422 (2002)
2. Al-Homidan, S., Wolkowicz, H.: Approximate and exact completion problems for Euclidean distance matrices using semidefinite programming. *Linear Algebra Appl.* **406**, 109–141 (2005)
3. Alfakih, A.Y., Anjos, M.F., Piccialli, V., Wolkowicz, H.: Euclidean distance matrices, semidefinite programming, and sensor network localization. *Port. Math.* **68**, 53–102 (2011)
4. Alfakih, A.Y., Khandani, A., Wolkowicz, H.: Solving Euclidean distance matrix completion problems via semidefinite programming. *Comput. Optim. Appl.* **12**, 13–30 (1999)
5. Alfakih, A.Y., Wolkowicz, H.: Matrix completion problems. In: Wolkowicz, H., Saigal, R., Vandenberghe, L. (eds.) *Handbook of Semidefinite Programming: Theory, Algorithms, and Applications*. Int. Ser. Oper. Res. Man. Sc., vol. 27, pp. 533–545. Kluwer Academic, Boston (2000)

6. Alfakih, A.Y.: Graph rigidity via Euclidean distance matrices. *Linear Algebra Appl.* **310**, 149–165 (2000)
7. Alfakih, A.Y.: On rigidity and realizability of weighted graphs. *Linear Algebra Appl.* **325**, 57–70 (2001)
8. Alfakih, A.Y.: On the uniqueness of Euclidean distance matrix completions. *Linear Algebra Appl.* **370**, 1–14 (2003)
9. Alfakih, A.Y.: On the uniqueness of Euclidean distance matrix completions: the case of points in general position. *Linear Algebra Appl.* **397**, 265–277 (2005)
10. Alfakih, A.Y.: On the nullspace, the rangespace and the characteristic polynomial of Euclidean distance matrices. *Linear Algebra Appl.* **416**, 348–354 (2006)
11. Alfakih, A.Y.: A remark on the faces of the cone of Euclidean distance matrices. *Linear Algebra Appl.* **414**, 266–270 (2006)
12. Alfakih, A.Y.: On dimensional rigidity of bar-and-joint frameworks. *Discrete Appl. Math.* **155**, 1244–1253 (2007)
13. Alfakih, A.Y., Wolkowicz, H.: Two theorems on Euclidean distance matrices and Gale transform. *Linear Algebra Appl.* **340**, 149–154 (2002)
14. Ames, B.P.W., Vavasis, S.A.: Nuclear norm minimization for the planted clique and biclique problems. <http://arxiv.org/abs/0901.3348> (2009). Accessed 21 Jan 2009
15. An, L.T.H., Tao, P.D.: Large-scale molecular optimization from distance matrices by a D.C. optimization approach. *SIAM J. Optim.* **14**, 77–114 (2003)
16. Aspnes, J., Eren, T., Goldenberg, D.K., Morse, A., Whiteley, W., Yang, Y.R., Anderson, B.D.O., Belhumeur, P.N.: A theory of network localization. *IEEE T. Mobile Comput.* **5**, 1663–1678 (2006)
17. Aspnes, J., Goldenberg, D., Yang, Y.R.: On the computational complexity of sensor network localization. *Lect. Notes Comput. Sc.* **3121**, 32–44 (2004)
18. Bădoiu, M., Demaine, E.D., Hajiaghayi, M., Indyk, P.: Low-dimensional embedding with extra information. *Discrete Comput. Geom.* **36**, 609–632 (2006)
19. Bakonyi, M., Johnson, C.: The Euclidean distance matrix completion problem. *SIAM J. Matrix Anal. Appl.* **16**, 646–654 (1995)
20. Barvinok, A.: Problems of distance geometry and convex properties of quadratic maps. *Discrete Comput. Geom.* **13**, 189–202 (1995)
21. Beck, A., Stoica, P., Li, J.: Exact and approximate solutions of source localization problems. *IEEE T. Signal Proces.* **56**, 1770–1778 (2008)
22. Beck, A., Teboulle, M., Chikishev, Z.: Iterative minimization schemes for solving the single source localization problem. *SIAM J. Optim.* **19**, 1397–1416 (2008)
23. Belk, M.: Realizability of graphs in three dimensions. *Discrete Comput. Geom.* **37**, 139–162 (2007)
24. Belk, M., Connelly, R.: Realizability of graphs. *Discrete Comput. Geom.* **37**, 125–137 (2007)
25. Biswas, P.: Semidefinite programming approaches to distance geometry problems. Ph.D. thesis, Stanford University (2007)
26. Biswas, P., Liang, T.-C., Toh, K.-C., Wang, T.-C., Ye, Y.: Semidefinite programming approaches for sensor network localization with noisy distance measurements, *IEEE T. Autom. Sci. Eng.* **3**, 360–371 (2006)
27. Biswas, P., Toh, K.-C., Ye, Y.: A distributed SDP approach for large-scale noisy anchor-free graph realization with applications to molecular conformation. *SIAM J. Sci. Comput.* **30**, 1251–1277 (2008)
28. Biswas, P., Ye, Y.: Semidefinite programming for ad hoc wireless sensor network localization. In: Information Processing in Sensor Networks, Berkeley, CA, 26–27 April 2004. Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, pp. 46–54. ACM, New York (2004)
29. Biswas, P., Ye, Y.: A distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization. In: Hager, W.W., Huang, S.-J., Pardalos, P.M., Prokopyev, O.A. (eds.) *Multiscale Optimization Methods and Applications. Nonconvex Optim. Appl.*, vol. 82, pp. 69–84. Springer, New York (2006)

30. Björck, Å.: Numerical Methods for Least Squares Problems. SIAM, Philadelphia (1996)
31. Blumenthal, L.M.: Theory and Applications of Distance Geometry, 2nd edn. Chelsea, New York (1970)
32. Bruck, J., Gao, J., Jiang, A.: Localization and routing in sensor networks by local angle information. *ACM Trans. Sen. Netw.* **5**, 1–31 (2009)
33. Bulusu, N., Heidemann, J., Estrin, D.: GPS-less low-cost outdoor localization for very small devices. *IEEE Pers. Commun.* **7**, 28–34 (2000)
34. Candès, E.J., Plan, Y.: Matrix completion with noise. *P. IEEE* **98**, 925–936 (2010)
35. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. *Found. Comput. Math.* **9**, 717–772 (2009)
36. Carter, M.W., Jin, H.H., Saunders, M.A., Ye, Y.: SpaseLoc: An adaptive subproblem algorithm for scalable wireless sensor network localization. *SIAM J. Optim.* **17**, 1102–1128 (2006)
37. Cassioli, A.: Solving the sensor network localization problem using an heuristic multi-stage approach. [http://www.optimization-online.org/DB\\_HTML/2009/03/2267.html](http://www.optimization-online.org/DB_HTML/2009/03/2267.html) (2009). Accessed 13 Jan 2010
38. Chua, C.B., Tunçel, L.: Invariance and efficiency of convex representations. *Math. Program.* **111**, 113–140 (2008)
39. Connelly, R.: Generic global rigidity. *Discrete Comput. Geom.* **33**, 549–563 (2005)
40. Costa, J.A., Patwari, N., Hero III, A.O.: Distributed weighted-multidimensional scaling for node localization in sensor networks. *ACM Trans. Sen. Netw.* **2**, 39–64 (2006)
41. Crippen, G.M.: Chemical distance geometry: Current realization and future projection. *J. Math. Chem.* **6**, 307–324 (1991)
42. Crippen, G.M., Havel, T.F.: Distance Geometry and Molecular Conformation. Chemometrics Series, vol. 15. Research Studies Press, Taunton (1988)
43. Critchley, F.: On certain linear mappings between inner-product and squared distance matrices. *Linear Algebra Appl.* **105**, 91–107 (1988)
44. Dattorro, J.: Convex Optimization & Euclidean Distance Geometry. Meboo Publishing USA (2008)
45. Ding, Y., Krislock, N., Qian, J., Wolkowicz, H.: Sensor network localization, Euclidean distance matrix completions, and graph realization. In: MobiCom Annual International Conference on Mobile Computing and Networking, San Francisco, CA, 14–19 September 2008. Proceedings of the First ACM International Workshop on Mobile Entity Localization and Tracking in GPS-Less Environment, pp. 129–134. ACM, New York (2008)
46. Ding, Y., Krislock, N., Qian, J., Wolkowicz, H.: Sensor network localization, Euclidean distance matrix completions, and graph realization. *Optim. Eng.* **11**, 45–66 (2010)
47. Doherty, L., Pister, K.S.J., El Ghaoui, L.: Convex position estimation in wireless sensor networks. In: IEEE INFOCOM, Anchorage, AK, 22–26 April 2001. Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, pp. 1655–1663 (2001)
48. Dong, Q., Wu, Z.: A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. *J. Global Optim.* **22**, 365–375 (2002)
49. Dong, Q., Wu, Z.: A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. *J. Global Optim.* **26**, 321–333 (2003)
50. dos Santos Carvalho, R., Lavor, C., Protti, F.: Extending the geometric build-up algorithm for the molecular distance geometry problem. *Inform. Process. Lett.* **108**, 234–237 (2008)
51. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. *Psychometrika* **1**, 211–218 (1936)
52. Emiris, I.Z., Nikitopoulos, T.G.: Molecular conformation search by distance matrix perturbations. *J. Math. Chem.* **37**, 233–253 (2005)
53. Eren, T., Goldenberg, O., Whiteley, W., Yang, Y.R., Morse, A.S., Anderson, B.D.O., Belhumeur, P.N.: Rigidity, computation, and randomization in network localization. In: IEEE INFOCOM, Hong Kong, 7–11 March 2004. Proceedings of the Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 4, pp. 2673–2684 (2004)

54. Farebrother, R.W.: Three theorems with applications to Euclidean distance matrices. *Linear Algebra Appl.* **95**, 11–16 (1987)
55. Fazel, M.: Matrix rank minimization with applications. Ph.D. thesis, Stanford University (2002)
56. Fazel, M., Hindi, H., Boyd, S.P.: Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices. In: American Control Conference, Denver, CO, 4–6 June 2003. Proceedings of the 2003 American Control Conference, vol. 3, pp. 2156–2162 (2003)
57. Fukuda, M., Kojima, M., Murota, K., Nakata, K.: Exploiting sparsity in semidefinite programming via matrix completion I: General framework. *SIAM J. Optim.* **11**, 647–674 (2001)
58. Glunt, W., Hayden, T., Raydan, M.: Molecular conformations from distance matrices. *J. Comput. Chem.* **14**, 114–120 (1993)
59. Glunt, W., Hayden, T.L., Hong, S., Wells, J.: An alternating projection algorithm for computing the nearest Euclidean distance matrix. *SIAM J. Matrix Anal. Appl.* **11**, 589–600 (1990)
60. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM* **42**, 1115–1145 (1995)
61. Goldfarb, D., Scheinberg, K.: Interior point trajectories in semidefinite programming. *SIAM J. Optim.* **8**, 871–886 (1998)
62. Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
63. Gower, J.C.: Euclidean distance geometry. *Math. Sci.* **7**, 1–14 (1982)
64. Gower, J.C.: Distance matrices and their Euclidean approximation. In: Data Analysis and Informatics, Versailles, 4–7 October 1983. Proceedings of the Third International Symposium on Data Analysis and Informatics, pp. 3–21. North-Holland, Amsterdam, (1984)
65. Gower, J.C.: Properties of Euclidean and non-Euclidean distance matrices. *Linear Algebra Appl.* **67**, 81–97 (1985)
66. Green, B.: The orthogonal approximation of an oblique structure in factor analysis. *Psychometrika* **17**, 429–440 (1952)
67. Grone, R., Johnson, C.R., Sá, E.M., Wolkowicz, H.: Positive definite completions of partial Hermitian matrices. *Linear Algebra Appl.* **58**, 109–124 (1984)
68. Grooms, I.G., Lewis, R.M., Trosset, M.W.: Molecular embedding via a second order dissimilarity parameterized approach. *SIAM J. Sci. Comput.* **31**, 2733–2756 (2009)
69. Havel, T.F., Kuntz, I.D., Crippen, G.M.: The theory and practice of distance geometry. *B. Math. Biol.* **45**, 665–720 (1983)
70. Havel, T.F.: Metric matrix embedding in protein structure calculations, NMR spectra analysis, and relaxation theory. *Magn. Reson. Chem.* **41**, S37–S50 (2003)
71. Hayden, T.L., Wells, J., Liu, W.M., Tarazaga, P.: The cone of distance matrices. *Linear Algebra Appl.* **144**, 153–169 (1991)
72. Hendrickson, B.: The molecule problem: Determining conformation from pairwise distances. Ph.D. thesis, Cornell University (1990)
73. Hendrickson, B.: Conditions for unique graph realizations. *SIAM J. Comput.* **21**, 65–84 (1992)
74. Hendrickson, B.: The molecule problem: Exploiting structure in global optimization. *SIAM J. Optim.* **5**, 835–857 (1995)
75. Higham, N.J.: Computing the polar decomposition—with applications. *SIAM J. Sci. Stat. Comp.* **7**, 1160–1174 (1986)
76. Jackson, B., Jordán, T.: Connected rigidity matroids and unique realizations of graphs. *J. Comb. Theory B* **94**, 1–29 (2005)
77. Jin, H.H.: Scalable sensor localization algorithms for wireless sensor networks. Ph.D. thesis, University of Toronto (2005)
78. Johnson, C.R., Tarazaga, P.: Connections between the real positive semidefinite and distance matrix completion problems. *Linear Algebra Appl.* **223/224**, 375–391 (1995). Special issue honoring Miroslav Fiedler and Vlastimil Pták.

79. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, IBM Thomas J. Watson Res. Center, Yorktown Heights, NY, 20–22 March 1972. Proceedings of a Symposium on the Complexity of Computer Computations, pp. 85–103. Plenum, New York (1972)
80. Kim, S., Kojima, M., Waki, H.: Exploiting sparsity in SDP relaxation for sensor network localization. SIAM J. Optim. **20**, 192–215 (2009)
81. Kim, S., Kojima, M., Waki, H., Yamashita, M.: SFSDP: a sparse version of full semidefinite programming relaxation for sensor network localization problems. Research Report B-457, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology (2009)
82. Krislock, N.: Semidefinite facial reduction for low-rank euclidean distance matrix completion. Ph.D. thesis, University of Waterloo (2010)
83. Krislock, N., Wolkowicz, H.: Explicit sensor network localization using semidefinite representations and facial reductions. SIAM J. Optim. **20**, 2679–2708 (2010)
84. Kumar, P.S., Madhavan, C.V.: Minimal vertex separators of chordal graphs. Discrete Appl. Math. **89**, 155–168 (1998)
85. Laurent, M.: A connection between positive semidefinite and Euclidean distance matrix completion problems. Linear Algebra Appl. **273**, 9–22 (1998)
86. Laurent, M.: A tour d’horizon on positive semidefinite and Euclidean distance matrix completion problems. In: Semidefinite Programming and Interior-Point Approaches for Combinatorial Optimization Problems, Fields Institute, Toronto, ON, 15–17 May 1996. Topics in Semidefinite and Interior-Point Methods (Fields Institute Communications), pp. 51–76. Amer. Math. Soc., Providence (1998)
87. Laurent, M.: Polynomial instances of the positive semidefinite and Euclidean distance matrix completion problems. SIAM J. Matrix Anal. Appl. **22**, 874–894 (2001)
88. Leung, N.-H. Z., Kim-Chuan Toh, K.-C.: An SDP-based divide-and-conquer algorithm for large-scale noisy anchor-free graph realization. SIAM J. Sci. Comput. **31**, 4351–4372 (2009)
89. Li, X.-Y.: Wireless Ad Hoc and Sensor Networks: Theory and Applications. Cambridge University Press (2008)
90. Megerian, S., Koushanfar, F., Potkonjak, M., Srivastava, M.B.: Worst and best-case coverage in sensor networks. IEEE T. Mobile Comput. **4**, 84–92 (2005)
91. Moore, D., Leonard, J., Rus, D., Teller, S.: Robust distributed network localization with noisy range measurements. In: SenSys’04 ACM Conference on Embedded Network Sensor Systems, Baltimore, MD, 3–5 November 2004. Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, pp. 50–61. ACM, New York (2004)
92. Moré, J.J., Wu, Z.: Global continuation for distance geometry problems. SIAM J. Optim. **7**, 814–836 (1997)
93. Moré, J.J., Wu, Z.: Distance geometry optimization for protein structures. J. Global Optim. **15**, 219–234 (1999)
94. Nawaz, S.: Anchor Free Localization for Ad-hoc Wireless Sensor Networks. Ph.D. thesis, University of New South Wales (2008)
95. Nie, J.: Sum of squares method for sensor network localization. Comput. Optim. Appl. **43**, 151–179 (2009)
96. Pong, T., Tseng, P.: (Robust) Edge-based semidefinite programming relaxation of sensor network localization. Math. Program. (2010). doi: 10.1007/s10107-009-0338-x
97. Ramana, M.V., Tunçel, L., Wolkowicz, H.: Strong duality for semidefinite programming. SIAM J. Optim. **7**, 641–662 (1997)
98. Recht, B., Fazel, M., Parrilo, P.A.: Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. SIAM Rev. **52**, 471–501 (2010)
99. Recht, B., Xu, W., Hassibi, B.: Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization. <http://arxiv.org/abs/0809.1260> (2008). Accessed 7 Sep 2008
100. Rockafellar, R.T.: Convex Analysis. Princeton University Press (1970)

101. Savvides, A., Han, C.-C., Strivastava, M.B.: Dynamic fine-grained localization in ad-hoc networks of sensors. In: MobiCom'01, Rome, Italy, 16–21 July 2001. Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, pp. 166–179. ACM, New York (2001)
102. Saxe, J.B.: Embeddability of weighted graphs in  $k$ -space is strongly NP-hard. In: Annual Allerton Conference on Communications, Control, and Computing, 10–12 October 1979. Proceedings of the 17th Allerton Conference on Communications, Control, and Computing, pp. 480–489 (1979)
103. Schneider, R.: Convex Bodies: The Brunn-Minkowski Theory. Cambridge University Press (1993)
104. Schoenberg, I.: Remarks to Maurice Fréchet's article “Sur la définition axiomatique d'une classe d'espace distanciés vectoriellement applicable sur l'espace de Hilbert”. *Ann. Math.* **36**, 724–732 (1935)
105. Schönemann, P.: A generalized solution of the orthogonal Procrustes problem. *Psychometrika* **31**, 1–10 (1966)
106. So, A.M.-C.: A semidefinite programming approach to the graph realization problem: theory, applications and extensions. Ph.D. thesis, Stanford University (2007)
107. So, A.M.-C., Ye, Y.: A semidefinite programming approach to tensegrity theory and realizability of graphs. In: SODA'06, Miami, FL, 22–24 January 2006. Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 766–775. ACM, New York (2006)
108. So, A.M.-C., Ye, Y.: Theory of semidefinite programming for sensor network localization. *Math. Program.* **109**, 367–384 (2007)
109. Stoyanova, T., Kerasiotis, F., Prayati, A., Papadopoulos, G.: Evaluation of impact factors on RSS accuracy for localization and tracking applications in sensor networks. *Telecommun. Syst.* **42**, 235–248 (2009)
110. Tarazaga, P.: Faces of the cone of Euclidean distance matrices: characterizations, structure and induced geometry. *Linear Algebra Appl.* **408**, 1–13 (2005)
111. Tarazaga, P., Hayden, T.L., Wells, J.: Circum-Euclidean distance matrices and faces. *Linear Algebra Appl.* **232**, 77–96 (1996)
112. Tseng, P.: Second-order cone programming relaxation of sensor network localization. *SIAM J. Optim.* **18**, 156–185 (2007)
113. Wang, Z., Zheng, S., Boyd, S., Ye, Y.: Further relaxations of the semidefinite programming approach to sensor network localization. *SIAM J. Optim.* **19**, 655–673 (2008)
114. Weinberger, K.Q., Sha, F., Saul, L.K.: Learning a kernel matrix for nonlinear dimensionality reduction. In: ICML'04: Banff, AB, 4–8 July 2004. Proceedings of the 21st International Conference on Machine Learning, p. 106–113. ACM, New York (2004)
115. Wu, D., Wu, Z.: An updated geometric build-up algorithm for solving the molecular distance geometry problems with sparse distance data. *J. Global Optim.* **37**, 661–673 (2007)
116. Wu, D., Wu, Z., Yuan, Y.: Rigid versus unique determination of protein structures with geometric buildup. *Optim. Lett.* **2**, 319–331 (2008)
117. Yang, Z., Liu, Y., Li, X.-Y.: Beyond trilateration: On the localizability of wireless ad-hoc networks. In: IEEE INFOCOM, Rio de Janeiro, Brazil, 19–25 April 2009. INFOCOM 2009, IEEE, pp. 2392–2400 (2009)
118. Yemini, Y.: Some theoretical aspects of position-location problems. In: 20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 29–31 October 1979. 20th Annual Symposium on Foundations of Computer Science, pp. 1–8 (1979)
119. Young, G., Householder, A.: Discussion of a set of points in terms of their mutual distances. *Psychometrika* **3**, 19–22 (1938)

# Chapter 31

## Sparse PCA: Convex Relaxations, Algorithms and Applications

Youwei Zhang, Alexandre d’Aspremont, and Laurent El Ghaoui

### 31.1 Introduction

Principal component analysis (PCA) is a classical tool for data analysis, visualization and dimensionality reduction and has a wide range of applications throughout science and engineering. Starting from a multivariate data set, PCA finds linear combinations of the variables called *principal components*, corresponding to orthogonal directions maximizing variance in the data. Numerically, a full PCA involves a singular value decomposition of the data matrix.

One of the key shortcomings of PCA is that the factors are linear combinations of *all* original variables; that is, most of factor coefficients (or loadings) are non-zero. This means that while PCA facilitates model interpretation and visualization by concentrating the information in a few factors, the factors themselves are still constructed using all variables, hence are often hard to interpret.

In many applications, the coordinate axes involved in the factors have a direct physical interpretation. In financial or biological applications, each axis might correspond to a specific asset or gene. In problems such as these, it is natural to seek a trade-off between the two goals of *statistical fidelity* (explaining most of the variance in the data) and *interpretability* (making sure that the factors involve only a few coordinate axes). Solutions that have only a few nonzero coefficients in the principal components are usually easier to interpret. Moreover, in some applications, nonzero coefficients have a direct cost (e.g., transaction costs in finance) hence there may be a direct trade-off between statistical fidelity and practicality. Our aim here

---

Y. Zhang (✉) • L. El Ghaoui  
EECS, University of California, Berkeley, CA 94720, USA  
e-mail: [zyw@eecs.berkeley.edu](mailto:zyw@eecs.berkeley.edu); [elghaoui@eecs.berkeley.edu](mailto:elghaoui@eecs.berkeley.edu)

A. d’Aspremont  
ORFE, Princeton University, Princeton, NJ 08544, USA  
e-mail: [aspremon@princeton.edu](mailto:aspremon@princeton.edu)

is to efficiently derive *sparse principal components*, i.e., a set of sparse vectors that explain a maximum amount of variance. Our motivation is that in many applications, the decrease in statistical fidelity required to obtain sparse factors is small and relatively benign.

In what follows, we will focus on the problem of finding sparse factors which explain a maximum amount of variance in the original data, which can be written

$$\max_{\|z\| \leq 1} z^T \Sigma z - \rho \mathbf{Card}(z) \quad (31.1)$$

in the variable  $z \in \mathbf{R}^n$ , where  $\Sigma \in \mathbf{S}_n$  is the (symmetric positive semi-definite) sample covariance matrix,  $\rho$  is a parameter controlling sparsity, and  $\mathbf{Card}(z)$  denotes the cardinality (or  $\ell_0$  norm) of  $z$ , i.e. the number of non zero coefficients of  $z$ .

While PCA is numerically easy, each factor requires computing a leading eigenvector, which can be done in  $O(n^2)$  floating point operations using the Lanczos method for example (see e.g. [12, Sects. 8.3, 9.1.1] or [30] for details), sparse PCA is a hard combinatorial problem. In fact, [22] show that the subset selection problem for ordinary least squares, which is NP-hard [25], can be reduced to a sparse generalized eigenvalue problem, of which sparse PCA is a particular instance. Sometimes factor rotation techniques are used to post-process the results from PCA and improve interpretability (see QUARTIMAX by [29], VARIMAX by [17] or [15] for a discussion). Results by e.g. [2] show that this naive approach has significantly worst convergence rates than the relaxations we present here. Another straightforward solution is to *threshold* to zero loadings with small magnitude [5], but outside of easy cases, the methods highlighted below always perform better in situation when only a few observations are available or when significant noise is present.

A more systematic approach to the problem arose in recent years, with various researchers proposing nonconvex algorithms (e.g., SCoTLASS by [16], SLRA by [34] or D.C. based methods [31] which find modified principal components with zero loadings. The SPCA algorithm, which is based on the representation of PCA as a regression-type optimization problem [33], allows the application of the LASSO [32], a penalization technique based on the  $\ell_1$  norm. With the exception of simple thresholding, all the algorithms above require solving non convex problems. Recently also, [9] derived an  $\ell_1$  based semidefinite relaxation for the sparse PCA problem (31.1) with a complexity of  $O(n^4 \sqrt{\log n})$  for a given  $\rho$ . [23] used greedy search and branch-and-bound methods to solve small instances of problem (31.1) exactly and get good solutions for larger ones. Each step of this greedy algorithm has complexity  $O(n^3)$ , leading to a total complexity of  $O(n^4)$  for a full set of solutions. [24] improve this bound in the regression/discrimination case. [14] use an extension of the power method to (locally) solve the problem defined here, as well as the “block” problem of finding several sparse principal components at once. Loss of orthogonality means that there is no natural method for deflating the matrix once a sparse principal component is found and [20] discusses several options, comparing the variance vs. orthogonality/sparsity tradeoffs they imply. Finally, [2]

derive explicit sample size thresholds for recovery of true sparse vector using either simple thresholding methods or semidefinite relaxations, in a spiked model for the covariance.

Here, we detail two semidefinite relaxations for sparse PCA, and describe algorithms to solve the relaxations efficiently. We also test these techniques on various data sets: newsgroup data, Senate voting records and stock market returns.

### 31.1.1 Notation

For a vector  $z \in \mathbf{R}$ , we let  $\|z\|_1 = \sum_{i=1}^n |z_i|$  and  $\|z\| = (\sum_{i=1}^n z_i^2)^{1/2}$ ,  $\mathbf{Card}(z)$  is the cardinality of  $z$ , i.e. the number of nonzero coefficients of  $z$ , while the support  $I$  of  $z$  is the set  $\{i : z_i \neq 0\}$  and we use  $I^c$  to denote its complement. For  $\beta \in \mathbf{R}$ , we write  $\beta_+ = \max\{\beta, 0\}$  and for  $X \in \mathbf{S}_n$  (the set of symmetric matrix of size  $n \times n$ ) with eigenvalues  $\lambda_i$ ,  $\mathbf{Tr}(X)_+ = \sum_{i=1}^n \max\{\lambda_i, 0\}$ . The vector of all ones is written  $\mathbf{1}$ , while the identity matrix is written  $\mathbf{I}$ . The diagonal matrix with the vector  $u$  on the diagonal is written  $\mathbf{diag}(u)$ .

## 31.2 Semidefinite Relaxations

Let  $\Sigma \in \mathbf{S}_n$  be a symmetric matrix. We consider the following sparse PCA problem

$$\phi(\rho) \equiv \max_{\|z\| \leq 1} z^T \Sigma z - \rho \mathbf{Card}(z) \quad (31.2)$$

in the variable  $z \in \mathbf{R}^n$  where  $\rho > 0$  is a parameter controlling sparsity. We assume without loss of generality that  $\Sigma \in \mathbf{S}_n$  is positive semidefinite and that the  $n$  variables are ordered by decreasing marginal variances, i.e. that  $\Sigma_{11} \geq \dots \geq \Sigma_{nn}$ . We also assume that we are given a square root  $A$  of the matrix  $\Sigma$  with  $\Sigma = A^T A$ , where  $A \in \mathbf{R}^{n \times n}$  and we denote by  $a_1, \dots, a_n \in \mathbf{R}^n$  the columns of  $A$ . Note that the problem and our algorithms are invariant by permutations of  $\Sigma$  and by the choice of square root  $A$ . In practice, we are very often given the data matrix  $A$  instead of the covariance  $\Sigma$ .

A problem that is directly related to (31.2) is that of computing a cardinality constrained maximum eigenvalue, by solving

$$\begin{aligned} & \text{maximize } z^T \Sigma z \\ & \text{subject to } \mathbf{Card}(z) \leq k \\ & \quad \|z\| = 1, \end{aligned} \quad (31.3)$$

in the variable  $z \in \mathbf{R}^n$ . Of course, this problem and (31.2) are related. By weak duality, an upper bound on the optimal value of (31.3) is given by

$$\inf_{\rho \in P} \phi(\rho) + \rho k .$$

where  $P$  is the set of penalty values for which  $\phi(\rho)$  has been computed. This means in particular that if a point  $z$  is provably optimal for (31.2), it is also globally optimum for (31.3) with  $k = \mathbf{Card}(z)$ .

### 31.2.1 A Semidefinite Relaxation with $\ell_1$ Penalization

Here, we briefly recall the  $\ell_1$  based relaxation derived in [9]. Following the *lifting procedure* for semidefinite relaxation described in [1, 18, 19] for example, we rewrite (31.3) as

$$\begin{aligned} & \text{maximize } \mathbf{Tr}(\Sigma X) \\ & \text{subject to } \mathbf{Tr}(X) = 1 \\ & \quad \mathbf{Card}(X) \leq k^2 \\ & \quad X \succeq 0, \mathbf{Rank}(X) = 1, \end{aligned} \tag{31.4}$$

in the (matrix) variable  $X \in \mathbf{S}^n$ . Programs (31.3) and (31.4) are equivalent, indeed if  $X$  is a solution to the above problem, then  $X \succeq 0$  and  $\mathbf{Rank}(X) = 1$  mean that we have  $X = xx^T$ , while  $\mathbf{Tr}(X) = 1$  implies that  $\|x\|_2 = 1$ . Finally, if  $X = xx^T$  then  $\mathbf{Card}(X) \leq k^2$  is equivalent to  $\mathbf{Card}(x) \leq k$ . We have made some progress by turning the convex maximization objective  $x^T \Sigma x$  and the nonconvex constraint  $\|x\|_2 = 1$  into a linear constraint and linear objective. Problem (31.4) is, however, still nonconvex and we need to relax both the rank and cardinality constraints.

Since for every  $u \in \mathbf{R}^n$ ,  $\mathbf{Card}(u) = q$  implies  $\|u\|_1 \leq \sqrt{q}\|u\|_2$ , we can replace the nonconvex constraint  $\mathbf{Card}(X) \leq k^2$ , by a weaker but convex constraint:  $\mathbf{1}^T |X| \mathbf{1} \leq k$ , where we exploit the property that  $\|X\|_F = \sqrt{x^T x} = 1$  when  $X = xx^T$  and  $\mathbf{Tr}(X) = 1$ . If we drop the rank constraint, we can form a relaxation of (31.4) and (31.3) as

$$\begin{aligned} & \text{maximize } \mathbf{Tr}(\Sigma X) \\ & \text{subject to } \mathbf{Tr}(X) = 1 \\ & \quad \mathbf{1}^T |X| \mathbf{1} \leq k \\ & \quad X \succeq 0, \end{aligned} \tag{31.5}$$

which is a semidefinite program in the variable  $X \in \mathbf{S}^n$ , where  $k$  is an integer parameter controlling the sparsity of the solution. The optimal value of this program will be an upper bound on the optimal value of the variational problem in (31.3). Here, the relaxation of  $\mathbf{Card}(X)$  in  $\mathbf{1}^T |X| \mathbf{1}$  corresponds to a classic technique which replaces the (non-convex) cardinality or  $l_0$  norm of a vector  $x$  with its largest convex lower bound on the unit box:  $|x|$ , the  $l_1$  norm of  $x$  (see [11] or [10] for other applications).

Problem (31.5) can be interpreted as a robust formulation of the maximum eigenvalue problem, with additive, componentwise uncertainty in the input matrix  $\Sigma$ . We again assume  $\Sigma$  to be symmetric and positive semidefinite. If we consider a variation in which we penalize by the  $\ell_1$  norm of the matrix  $X$  instead of imposing a hard bound, to get

$$\begin{aligned} & \text{maximize} && \mathbf{Tr}(\Sigma X) - \rho \mathbf{1}^T |X| \mathbf{1} \\ & \text{subject to} && \mathbf{Tr}(X) = 1 \\ & && X \succeq 0, \end{aligned} \quad (31.6)$$

which is a semidefinite program in the variable  $X \in \mathbf{S}^n$ , where  $\rho > 0$  controls the magnitude of the penalty. We can rewrite this problem as

$$\max_{X \succeq 0, \mathbf{Tr}(X)=1} \min_{|U_{ij}| \leq \rho} \mathbf{Tr}(X(\Sigma + U)) \quad (31.7)$$

in the variables  $X \in \mathbf{S}^n$  and  $U \in \mathbf{S}^n$ . This yields the following dual to (31.6)

$$\begin{aligned} & \text{minimize} && \lambda^{\max}(\Sigma + U) \\ & \text{subject to} && |U_{ij}| \leq \rho, \quad i, j = 1, \dots, n, \end{aligned} \quad (31.8)$$

which is a maximum eigenvalue problem with variable  $U \in \mathbf{S}^n$ . This gives a natural robustness interpretation to the relaxation in (31.6): it corresponds to a worst-case maximum eigenvalue computation, with componentwise bounded noise of intensity  $\rho$  imposed on the matrix coefficients.

Finally, the KKT conditions (see [4, Sect. 5.9.2]) for problem (31.6) and (31.8) are given by

$$\begin{cases} (\Sigma + U)X = \lambda^{\max}(\Sigma + U)X \\ U \circ X = -\rho |X| \\ \mathbf{Tr}(X) = 1, \quad X \succeq 0 \\ |U_{ij}| \leq \rho, \quad i, j = 1, \dots, n. \end{cases} \quad (31.9)$$

If the eigenvalue  $\lambda^{\max}(\Sigma + U)$  is simple (when, for example,  $\lambda^{\max}(A)$  is simple and  $\rho$  is sufficiently small), the first condition means that  $\mathbf{Rank}(X) = 1$  and the semidefinite relaxation is *tight*, with in particular  $\mathbf{Card}(X) = \mathbf{Card}(x)^2$  if  $x$  is the dominant eigenvector of  $X$ . When the optimal solution  $X$  is not of rank one because of degeneracy (i.e. when  $\lambda^{\max}(\Sigma + U)$  has multiplicity strictly larger than one), we can truncate  $X$  as in [1, 18], retaining only the dominant eigenvector  $x$  as an approximate solution to the original problem. In that degenerate scenario however, the dominant eigenvector of  $X$  is not guaranteed to be as sparse as the matrix itself.

### 31.2.2 A Semidefinite Relaxation with $\ell_0$ Penalization

We summarize here the results in [7]. We begin by reformulating (31.2) as a relatively simple convex maximization problem. Suppose that  $\rho \geq \Sigma_{11}$ . Since  $z^T \Sigma z \leq \Sigma_{11} (\sum_{i=1}^n |z_i|)^2$  and  $(\sum_{i=1}^n |z_i|)^2 \leq \|z\|^2 \mathbf{Card}(z)$  for all  $z \in \mathbf{R}^n$ , we have

$$\begin{aligned}\phi(\rho) &= \max_{\|z\| \leq 1} z^T \Sigma z - \rho \mathbf{Card}(z) \\ &\leq (\Sigma_{11} - \rho) \mathbf{Card}(z) \\ &\leq 0,\end{aligned}$$

hence the optimal solution to (31.2) when  $\rho \geq \Sigma_{11}$  is  $z = 0$ . From now on, we assume  $\rho \leq \Sigma_{11}$  in which case the inequality  $\|z\| \leq 1$  is tight. We can represent the sparsity pattern of a vector  $z$  by a vector  $u \in \{0, 1\}^n$  and rewrite (31.2) in the equivalent form

$$\begin{aligned}\phi(\rho) &= \max_{u \in \{0, 1\}^n} \lambda_{\max}(\mathbf{diag}(u) \Sigma \mathbf{diag}(u)) - \rho \mathbf{1}^T u \\ &= \max_{u \in \{0, 1\}^n} \lambda_{\max}(\mathbf{diag}(u) A^T A \mathbf{diag}(u)) - \rho \mathbf{1}^T u \\ &= \max_{u \in \{0, 1\}^n} \lambda_{\max}(A \mathbf{diag}(u) A^T) - \rho \mathbf{1}^T u,\end{aligned}$$

using the fact that  $\mathbf{diag}(u)^2 = \mathbf{diag}(u)$  for all variables  $u \in \{0, 1\}^n$  and that for any matrix  $B$ ,  $\lambda_{\max}(B^T B) = \lambda_{\max}(BB^T)$ . We then have

$$\begin{aligned}\phi(\rho) &= \max_{u \in \{0, 1\}^n} \lambda_{\max}(A \mathbf{diag}(u) A^T) - \rho \mathbf{1}^T u \\ &= \max_{\|x\|=1} \max_{u \in \{0, 1\}^n} x^T A \mathbf{diag}(u) A^T x - \rho \mathbf{1}^T u \\ &= \max_{\|x\|=1} \max_{u \in \{0, 1\}^n} \sum_{i=1}^n u_i ((a_i^T x)^2 - \rho).\end{aligned}$$

Hence we finally get, after maximizing in  $u$  (and using  $\max_{v \in \{0, 1\}} \beta v = \beta_+$ )

$$\phi(\rho) = \max_{\|x\|=1} \sum_{i=1}^n ((a_i^T x)^2 - \rho)_+, \quad (31.10)$$

which is a nonconvex problem in the variable  $x \in \mathbf{R}^n$ . We then select variables  $i$  such that  $(a_i^T x)^2 - \rho > 0$ . Note that if  $\Sigma_{ii} = a_i^T a_i < \rho$ , we must have  $(a_i^T x)^2 \leq \|a_i\|^2 \|x\|^2 < \rho$  hence variable  $i$  will never be part of the optimal subset and we can remove it.

Because the variable  $x$  appears solely through  $X = xx^T$ , we can reformulate the problem in terms of  $X$  only, using the fact that when  $\|x\| = 1$ ,  $X = xx^T$  is equivalent to  $\mathbf{Tr}(X) = 1$ ,  $X \succeq 0$  and  $\mathbf{Rank}(X) = 1$ . We thus rewrite (31.10) as

$$\begin{aligned}\phi(\rho) &= \max. \sum_{i=1}^n (a_i^T X a_i - \rho)_+ \\ \text{s.t. } & \mathbf{Tr}(X) = 1, \mathbf{Rank}(X) = 1 \\ & X \succeq 0.\end{aligned}$$

Note that because we are maximizing a convex function over  $\mathcal{A}_n = \{X \in \mathbf{S}_n : \mathbf{Tr}(X) = 1, X \geq 0\}$  which is convex, the solution must be an extreme point of  $\mathcal{A}_n$  (i.e. a rank one matrix), hence we can drop the rank constraint here. Unfortunately,  $X \mapsto (a_i^T X a_i - \rho)_+$ , the function we are *maximizing*, is convex in  $X$  and not concave, which means that the above problem is still hard. However, we show below that on rank one elements of  $\mathcal{A}_n$ , it is also equal to a concave function of  $X$ , and we use this to produce a semidefinite relaxation of problem (31.2).

**Proposition 31.1.** *Let  $A \in \mathbf{R}^{n \times n}$ ,  $\rho \geq 0$  and denote by  $a_1, \dots, a_n \in \mathbf{R}^n$  the columns of  $A$ , an upper bound on*

$$\begin{aligned} \phi(\rho) &= \max. \sum_{i=1}^n (a_i^T X a_i - \rho)_+ \\ \text{s.t. } &\mathbf{Tr}(X) = 1, X \geq 0, \mathbf{Rank}(X) = 1 \end{aligned} \quad (31.11)$$

can be computed by solving

$$\begin{aligned} \psi(\rho) &= \max. \sum_{i=1}^n \mathbf{Tr}(X^{1/2} B_i X^{1/2})_+ \\ \text{s.t. } &\mathbf{Tr}(X) = 1, X \geq 0. \end{aligned} \quad (31.12)$$

in the variables  $X \in \mathbf{S}_n$ , where  $B_i = a_i a_i^T - \rho \mathbf{I}$ , or also

$$\begin{aligned} \psi(\rho) &= \max. \sum_{i=1}^n \mathbf{Tr}(P_i B_i) \\ \text{s.t. } &\mathbf{Tr}(X) = 1, X \geq 0, X \geq P_i \geq 0, \end{aligned} \quad (31.13)$$

which is a semidefinite program in the variables  $X \in \mathbf{S}_n$ ,  $P_i \in \mathbf{S}_n$ .

*Proof.* We let  $X^{1/2}$  be the positive square root (i.e. with nonnegative eigenvalues) of a symmetric positive semi-definite matrix  $X$ . In particular, if  $X = xx^T$  with  $\|x\| = 1$ , then  $X^{1/2} = X = xx^T$ , and for all  $\beta \in \mathbf{R}$ ,  $\beta xx^T$  has one eigenvalue equal to  $\beta$  and  $n-1$  equal to 0, which implies  $\mathbf{Tr}(\beta xx^T)_+ = \beta_+$ . We thus get

$$\begin{aligned} (a_i^T X a_i - \rho)_+ &= \mathbf{Tr}\left((a_i^T x x^T a_i - \rho)x x^T\right)_+ \\ &= \mathbf{Tr}\left(x(a_i^T a_i x^T - \rho)x^T\right)_+ \\ &= \mathbf{Tr}\left(X^{1/2} a_i a_i^T X^{1/2} - \rho X\right)_+ = \mathbf{Tr}\left(X^{1/2} (a_i a_i^T - \rho \mathbf{I}) X^{1/2}\right)_+. \end{aligned}$$

For any symmetric matrix  $B$ , the function  $X \mapsto \mathbf{Tr}(X^{1/2} B X^{1/2})_+$  is concave on the set of symmetric positive semidefinite matrices, because we can write it as

$$\begin{aligned} \mathbf{Tr}\left(X^{1/2} B X^{1/2}\right)_+ &= \max_{\{0 \leq P \leq X\}} \mathbf{Tr}(PB) \\ &= \min_{\{Y \geq B, Y \geq 0\}} \mathbf{Tr}(YX), \end{aligned}$$

where this last expression is a concave function of  $X$  as a pointwise minimum of affine functions. We can now relax the original problem into a convex optimization problem by simply dropping the rank constraint, to get

$$\begin{aligned}\psi(\rho) \equiv & \max. \sum_{i=1}^n \text{Tr}(X^{1/2} a_i a_i^T X^{1/2} - \rho X)_+ \\ \text{s.t. } & \text{Tr}(X) = 1, X \geq 0,\end{aligned}$$

which is a convex program in  $X \in \mathbf{S}_n$ . Note that because  $B_i$  has at most one nonnegative eigenvalue, we can replace  $\text{Tr}(X^{1/2} a_i a_i^T X^{1/2} - \rho X)_+$  by  $\lambda_{\max}(X^{1/2} a_i a_i^T X^{1/2} - \rho X)_+$  in the above program. Using the representation of  $\text{Tr}(X^{1/2} B_i X^{1/2})_+$  detailed above, problem (31.12) can be written as a semidefinite program

$$\begin{aligned}\psi(\rho) = & \max. \sum_{i=1}^n \text{Tr}(P_i B_i) \\ \text{s.t. } & \text{Tr}(X) = 1, X \geq 0, X \geq P_i \geq 0,\end{aligned}$$

in the variables  $X \in \mathbf{S}_n$ ,  $P_i \in \mathbf{S}_n$ , which is the desired result.  $\square$

Note that we always have  $\psi(\rho) \geq \phi(\rho)$  and when the solution to the above semidefinite program has rank one,  $\psi(\rho) = \phi(\rho)$  and the semidefinite relaxation (31.13) is *tight*. This simple fact allows us to derive sufficient global optimality conditions for the original sparse PCA problem. We recall in particular the following result from [7] which provides sufficient conditions for a particular nonzero coefficient pattern  $I$  to be globally optimal. The optimal solution  $x$  to (31.2) is then found by solving an eigenvalue problem on the principal submatrix of  $\Sigma$  with support  $I$ .

**Proposition 31.2.** *Let  $A \in \mathbf{R}^{n \times n}$ ,  $\rho \geq 0$ ,  $\Sigma = A^T A$  with  $a_1, \dots, a_n \in \mathbf{R}^n$  the columns of  $A$ . Given a sparsity pattern  $I$ , setting  $x$  to be the largest eigenvector of  $\sum_{i \in I} a_i a_i^T$ , if there is a  $\rho^* \geq 0$  such that the following conditions hold*

$$\max_{i \in I^c} (a_i^T x)^2 < \rho^* < \min_{i \in I} (a_i^T x)^2 \quad \text{and} \quad \lambda_{\max} \left( \sum_{i=1}^n Y_i \right) \leq \sum_{i \in I} \left( (a_i^T x)^2 - \rho^* \right),$$

with the dual variables  $Y_i$  defined as

$$Y_i = \max \left\{ 0, \rho \frac{(a_i^T a_i - \rho)}{(\rho - (a_i^T x)^2)} \right\} \frac{(\mathbf{I} - x x^T) a_i a_i^T (\mathbf{I} - x x^T)}{\|(\mathbf{I} - x x^T) a_i\|^2}, \quad \text{when } i \in I^c,$$

and

$$Y_i = \frac{B_i x x^T B_i}{x^T B_i x}, \quad \text{when } i \in I,$$

then the sparsity pattern  $I$  is globally optimal for the sparse PCA problem (31.2) with  $\rho = \rho^*$  and we can form an optimal solution  $z$  by solving the maximum eigenvalue problem

$$z = \underset{\{z_{I^c}=0, \|z\|=1\}}{\operatorname{argmax}} z^T \Sigma z.$$

This result also provides tractable *lower bounds* on the optimal value of (31.2) whenever the solution is not optimal.

### 31.3 Algorithms

In this section, we describe algorithms for solving the semidefinite relaxations detailed above. We also describe greedy methods to improve the quality of these solutions.

#### 31.3.1 First-Order Methods

Again, given a covariance matrix  $\Sigma \in \mathbf{S}_n$ , the DSPCA code solves a penalized formulation of problem (31.5), written as

$$\begin{aligned} & \text{maximize } \text{Tr}(\Sigma X) - \rho \mathbf{1}^T |X| \mathbf{1} \\ & \text{subject to } \text{Tr}(X) = 1 \\ & \quad X \geq 0, \end{aligned} \tag{31.14}$$

in the variable  $X \in \mathbf{S}_n$ . The dual of this program can be written as

$$\begin{aligned} & \text{minimize } f(U) = \lambda^{\max}(\Sigma + U) \\ & \text{subject to } |U_{ij}| \leq \rho. \end{aligned} \tag{31.15}$$

in the variable  $U \in \mathbf{S}^n$ . The algorithm in [9, 28] regularizes the objective  $f(U)$  in (31.15), replacing it by the smooth (i.e. with Lipschitz continuous gradient) uniform approximation

$$f_\mu(U) = \mu \log(\text{Tr} \exp((\Sigma + U)/\mu)) - \mu \log n.$$

Following [26], solving the smooth problem

$$\min_{U \in Q} f_\mu(U)$$

where  $Q = \{U \in \mathbf{S}^n, |U_{ij}| \leq \rho\}$ , with  $\mu = \epsilon/2 \log(n)$  then produces an  $\epsilon$ -approximate solution to (31.14). The key difference between the minimization scheme developed in [26] and classical gradient minimization methods is that it is not a descent method but achieves a complexity of  $O(L/N^2)$  instead of  $O(1/N)$  for gradient descent, where  $N$  is the number of iterations and  $L$  the Lipschitz constant of the gradient. Furthermore, this convergence rate is provably optimal for this particular class of convex minimization problems (see [27, Theorem 2.1.13]). Thus, by sacrificing the (local) properties of descent directions, we improve the (global) complexity estimate by an order of magnitude. For our problem here, once the regularization parameter  $\mu$  is set, the algorithm is detailed as Algorithm 2.

The algorithm has four main steps. Step one computes the (smooth) function value and gradient. The second step computes the *gradient mapping*, which matches

**Algorithm 2** First-Order Algorithm.

---

**Input:** The covariance  $\Sigma \in \mathbf{R}^{n \times n}$ , and a parameter  $\rho > 0$  controlling sparsity.

- 1: **for**  $i = 1$  to  $N$  **do**
- 2:   Compute  $f_\mu(U_i)$  and  $\nabla f_\mu(U_i)$
- 3:   Find  $Y_i = \arg \min_{Y \in Q} \langle \nabla f_\mu(U_i), Y \rangle + \frac{1}{2} L \|U_i - Y\|_F^2$
- 4:   Find  $W_i = \arg \min_{W \in Q} \left\{ \frac{L\|W\|_F^2}{2} + \sum_{j=0}^N \frac{j+1}{2} (f_\mu(U_j) + \langle \nabla f_\mu(U_j), W - U_j \rangle) \right\}$
- 5:   Set  $U_{i+1} = \frac{2}{i+3} W_i + \frac{i+1}{i+3} Y_i$
- 6: **end for**

**Output:** A matrix  $U \in \mathbf{S}_n$ .

---

the gradient step for unconstrained problems (see [27, p.86]). Step three and four update an *estimate sequence* see ([27, p.72]) of  $f_\mu$  whose minimum can be computed explicitly and gives an increasingly tight upper bound on the minimum of  $f_\mu$ . We now present these steps in detail for our problem (we write  $U$  for  $U_i$  and  $X$  for  $X_i$ ).

*Step 1.* The most expensive step in the algorithm is the first, the computation of  $f_\mu$  and its gradient. The function value can be reliably computed as

$$f_\mu(U) = d_{\max} + \mu \log \left( \sum_{i=1}^n \exp \left( \frac{d_i - d_{\max}}{\mu} \right) \right) - \mu \log n .$$

where  $d_i$  are the eigenvalues of  $\Sigma + U$ . The gradient  $\nabla f_\mu(U)$  can be computed explicitly as

$$\nabla f_\mu(U) := \exp((\Sigma + U)/\mu) / \mathbf{Tr}(\exp((\Sigma + U)/\mu)) .$$

which means computing the same matrix exponential.

*Step 2.* This step involves a problem of the form

$$\arg \min_{Y \in Q} \langle \nabla f_\mu(U), Y \rangle + \frac{1}{2} L \|U - Y\|_F^2 ,$$

where  $U$  is given. The above problem can be reduced to a Euclidean projection

$$\arg \min_{\|Y\|_\infty \leq 1} \|Y - V\|_F , \quad (31.16)$$

where  $V = U - L^{-1} \nabla f_\mu(U)$  is given. The solution is given by

$$Y_{ij} = \mathbf{sgn}(V_{ij}) \min(|V_{ij}|, 1), \quad i, j = 1, \dots, n .$$

*Step 3.* The third step involves solving a Euclidean projection problem similar to (31.16), with the solution  $V$  defined by

$$V = -\frac{1}{L} \sum_{i=0}^k \frac{i+1}{2} \nabla f_\mu(U_i).$$

### 31.3.1.1 Stopping Criterion

We can stop the algorithm when the duality gap is smaller than  $\epsilon$ :

$$\text{gap}_k = \lambda_{\max}(\Sigma + U_k) - \mathbf{Tr} \Sigma X_i + \mathbf{1}^T |X_i| \mathbf{1} \leq \epsilon,$$

where  $X_k = \nabla f_\mu(U)$  is our current estimate of the dual variable. The above gap is necessarily non-negative, since both  $X_i$  and  $U_i$  are feasible for the primal and dual problem, respectively. This is checked periodically, for example every 100 iterations.

### 31.3.1.2 Complexity

Overall the algorithm requires

$$O\left(\rho \frac{n \sqrt{\log n}}{\epsilon}\right) \quad (31.17)$$

iterations [9, 28]. The main step at each iteration is computing the matrix exponential  $\exp((\Sigma + U)/\mu)$  (see [21] for a comprehensive survey) at a cost of  $O(n^3)$  flops.

### 31.3.2 Greedy Methods

We can also find good solution to problem (31.2), or improve existing solutions, using greedy methods. We first present very simple preprocessing solutions with complexity  $O(n \log n)$  and  $O(n^2)$ . We then recall a simple greedy algorithm with complexity  $O(n^4)$ . Finally, our first contribution in this section is to derive an approximate greedy algorithm that computes a full set of (approximate) solutions for problem (31.2), with complexity  $O(n^3)$ .

**Algorithm 3** Greedy Search Algorithm.**Input:**  $\Sigma \in \mathbf{R}^{n \times n}$ 

- 1: Preprocessing: sort variables by decreasing diagonal elements and permute elements of  $\Sigma$  accordingly.
- 2: Compute the Cholesky decomposition  $\Sigma = A^T A$ .
- 3: Initialization:  $I_1 = \{1\}$ ,  $x_1 = a_1 / \|a_1\|$ .
- 4: **for**  $i = 1$  to  $k^{\text{target}}$  **do**
- 5:   Compute  $i_k = \operatorname{argmax}_{i \notin I_k} \lambda_{\max} \left( \sum_{j \in I_k \cup \{i\}} a_j a_j^T \right)$ .
- 6:   Set  $I_{k+1} = I_k \cup \{i_k\}$  and compute  $x_{k+1}$  as the leading eigenvector of  $\sum_{j \in I_{k+1}} a_j a_j^T$ .
- 7: **end for**

**Output:** Sparsity patterns  $I_k$ .**31.3.2.1 Sorting and Thresholding**

The simplest ranking algorithm is to sort the diagonal of the matrix  $\Sigma$  and rank the variables by variance. This works intuitively because the diagonal is a rough proxy for the eigenvalues: the Schur–Horn theorem states that the diagonal of a matrix majorizes its eigenvalues [13]; sorting costs  $O(n \log n)$ . Another quick solution is to compute the leading eigenvector of  $\Sigma$  and form a sparse vector by thresholding to zero the coefficients whose magnitude is smaller than a certain level. This can be done with cost  $O(n^2)$ .

**31.3.2.2 Full Greedy Solution**

Following [23], starting from an initial solution of cardinality one at  $\rho = \Sigma_{11}$ , we can update an increasing sequence of index sets  $I_k \subseteq [1, n]$ , scanning all the remaining variables to find the index with maximum variance contribution.

At every step,  $I_k$  represents the set of nonzero elements (or sparsity pattern) of the current point and we can define  $z_k$  as the solution to problem (31.2) given  $I_k$ , which is:

$$z_k = \operatorname{argmax}_{\{z_{I_k^c} = 0, \|z\|=1\}} z^T \Sigma z - \rho k,$$

which means that  $z_k$  is formed by padding zeros to the leading eigenvector of the submatrix  $\Sigma_{I_k, I_k}$ . Note that the entire algorithm can be written in terms of a factorization  $\Sigma = A^T A$  of the matrix  $\Sigma$ , which means significant computational savings when  $\Sigma$  is given as a Gram matrix. The matrices  $\Sigma_{I_k, I_k}$  and  $\sum_{i \in I_k} a_i a_i^T$  have the same eigenvalues and if  $z$  is an eigenvector of  $\Sigma_{I_k, I_k}$ , then  $A_{I_k} z / \|A_{I_k} z\|$  is an eigenvector of  $A_{I_k} A_{I_k}^T$ .

**Algorithm 4** Approximate Greedy Search Algorithm.**Input:**  $\Sigma \in \mathbf{R}^{n \times n}$ 

- 1: Preprocessing: sort variables by decreasing diagonal elements and permute elements of  $\Sigma$  accordingly.
- 2: Compute the Cholesky decomposition  $\Sigma = A^T A$ .
- 3: Initialization:  $I_1 = \{1\}$ ,  $x_1 = a_1 / \|a_1\|$ .
- 4: **for**  $i = 1$  to  $k^{\text{target}}$  **do**
- 5:   Compute  $i_k = \operatorname{argmax}_{i \notin I_k} (x_k^T a_i)^2$ .
- 6:   Set  $I_{k+1} = I_k \cup \{i_k\}$  and compute  $x_{k+1}$  as the leading eigenvector of  $\sum_{j \in I_{k+1}} a_j a_j^T$ .
- 7: **end for**

**Output:** Sparsity patterns  $I_k$ .**31.3.2.3 Approximate Greedy Solution**

Computing  $n - k$  eigenvalues at each iteration is costly and we can use the fact that  $uu^T$  is a subgradient of  $\lambda_{\max}$  at  $X$  if  $u$  is a leading eigenvector of  $X$  [4], to get:

$$\lambda_{\max} \left( \sum_{j \in I_k \cup \{i\}} a_j a_j^T \right) \geq \lambda_{\max} \left( \sum_{j \in I_k} a_j a_j^T \right) + (x_k^T a_i)^2, \quad (31.18)$$

which means that the variance is increasing by at least  $(x_k^T a_i)^2$  when variable  $i$  is added to  $I_k$ . This provides a lower bound on the objective which does not require finding  $n - k$  eigenvalues at each iteration. We then derive the following algorithm.

Again, at every step,  $I_k$  represents the set of nonzero elements (or sparsity pattern) of the current point and we can define  $z_k$  as the solution to problem (31.2) given  $I_k$ , which is:

$$z_k = \operatorname{argmax}_{\{z_{I_k^c} = 0, \|z\|=1\}} z^T \Sigma z - \rho k,$$

which means that  $z_k$  is formed by padding zeros to the leading eigenvector of the submatrix  $\Sigma_{I_k, I_k}$ . Better points can be found by testing the variables corresponding to the  $p$  largest values of  $(x_k^T a_i)^2$  instead of picking only the best one.

**31.3.2.4 Computational Complexity**

The complexity of computing a greedy regularization path using the classic greedy algorithm in Sect. 31.3.2.2 is  $O(n^4)$ : at each step  $k$ , it computes  $(n - k)$  maximum eigenvalue of matrices with size  $k$ . The approximate algorithm in Sect. 31.3.2.3 computes a full path in  $O(n^3)$ : the first Cholesky decomposition is  $O(n^3)$ , while the complexity of the  $k$ -th iteration is  $O(k^2)$  for the maximum eigenvalue problem and  $O(n^2)$  for computing all products  $(x_k^T a_j)$ . Also, when the matrix  $\Sigma$  is directly given as a Gram matrix  $A^T A$  with  $A \in \mathbf{R}^{q \times n}$  with  $q < n$ , it is advantageous to use  $A$  directly as the square root of  $\Sigma$  and the total complexity of getting the path up to cardinality  $p$  is then reduced to  $O(p^3 + p^2 n)$  (which is  $O(p^3)$  for the eigenvalue problems and  $O(p^2 n)$  for computing the vector products).

## 31.4 Applications

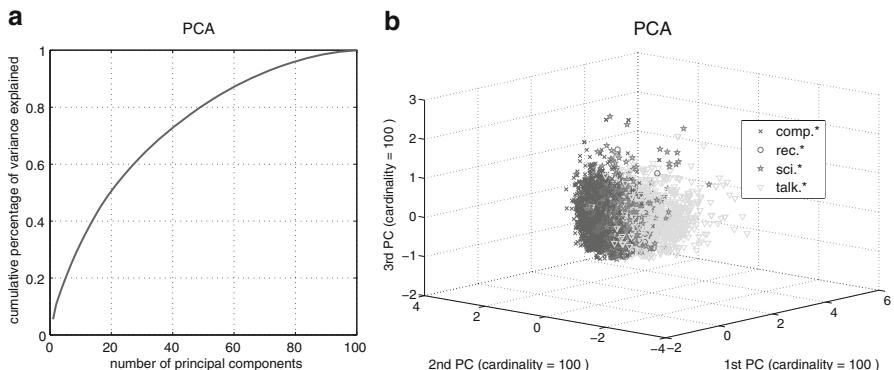
In this section, we illustrate the sparse PCA approach in applications: in news (text), finance and voting data from the US Senate.

### 31.4.1 News Data

Our first dataset is a small version of the “20-newsgroups” data.<sup>1</sup> The data records binary occurrences of 100 specific words across 16,242 postings, where the postings have been tagged by the highest level domain in Usenet.

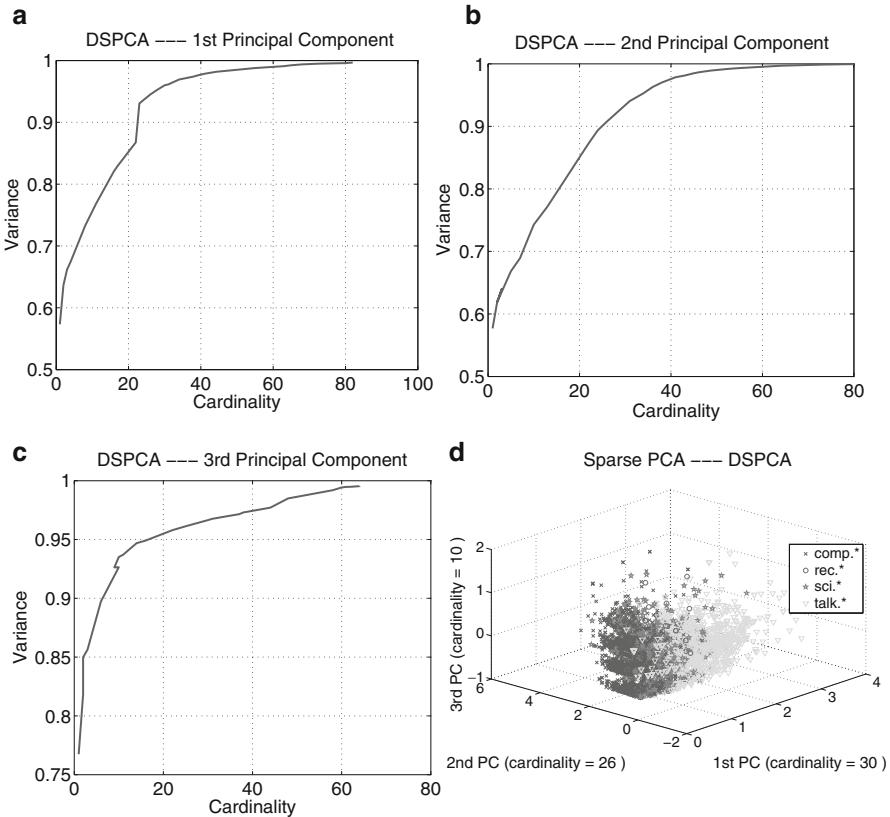
Each posting is viewed as one point in a 100-dimensional space. We begin with a standard PCA on the data. Figure 31.1 (left) shows the cumulative percentage of variance explained as we increase the number of principal components. The slow increase means that the data does not lie within a subspace of significantly low dimension. We can anyway proceed to visualize the data: Fig. 31.1 (right) is the result obtained by projecting it on a subspace of dimension 3, chosen by selecting the eigenvectors corresponding to the three largest eigenvalues of the covariance matrix. Since these 3 vectors are dense, the axes in Fig. 31.1 (right) do not have a clear interpretation.

With sparse PCA, we hope to find a set of corresponding sparse principal components, which still help with visualization nearly as well as PCA does, and yet reveal some interesting structure. To achieve this, we have run the first-order algorithm of Sect. 31.3.1 (referred to as “DSPCA” hereafter) on the data with a range of values for the penalty parameter  $\rho$ . We obtained a plot of the variance explained



**Fig. 31.1** PCA with 20-Newsgroups data. *Left:* Explained variance vs. number of PCs. *Right:* 3D visualization via PCA

<sup>1</sup>available from <http://cs.nyu.edu/~roweis/data.html>.



**Fig. 31.2** Sparse PCA on the 20Newsgroups data set. First three principal components and 3D visualization. The first three principal components have cardinalities 26, 30 and 10 respectively

by the first sparse principal component (PC), as a function of its cardinality (Fig. 31.2). We then selected a cardinality that can explain at least 90% of the variance explained by the first principal component obtained from PCA. Then we have deflated the covariance matrix by taking out the part due to the first sparse PC, and then repeated the above procedure to obtain the second sparse PC. In the same way, we have solved for the third sparse PC. Figure 31.2 also shows the projection of the data on the 3-dimensional subspace that is spanned by the three sparse PCs obtained above.

We first note that only a small number of words, out of the total of 100 words that can appear in each sparse PC, can explain more than 90% of variance explained by the corresponding PC. Specifically, we obtain 30 words for the first PC, 26 for the second, and 10 for the third. The lists of words associated with each sparse PCs is given in Table 31.1, and reveals some structure about each one of the sparse PCs. That is, the 30 words associated with the first sparse PC are almost all about politics and religion, the 26 words in the second sparse PC are all computer-related, and the

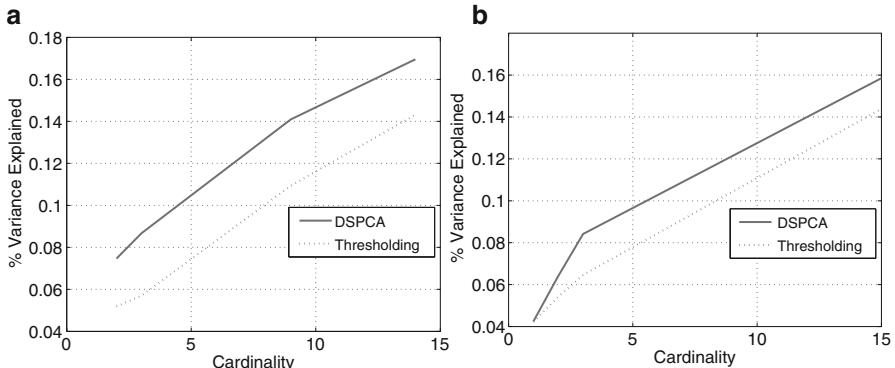
**Table 31.1** Words associated with the first three sparse PCs

1st PC (30 words)	2nd PC (26 words)	3rd PC (10 words)
Fact	Help	Problem
Question	Problem	University
World	System	Email
Course	Email	State
Case	Windows	Research
Problem	Program	Science
God	Computer	Phone
Government	Software	World
Human	University	Fact
State	Version	Question
Number	Files	
Christian	Drive	
Evidence	Data	
Law	Card	
Power	Dos	
Religion	God	
Children	Disk	
Jesus	Pc	
System	Graphics	
Rights	Ftp	
War	Memory	
Jews	Christian	
Help	Phone	
Bible	Video	
Earth	Fact	
Science	Display	
Research		
Israel		
President		
Gun		

majority of the 10 words in the third sparse PC concerns science. Hence, applying sparse PCA to this data set allows to discover structure that is otherwise hidden in the standard PCA, for example that the first principal component is mainly related to politics and religion.

We also run the thresholded PCA and DSPCA algorithms over the collection of 1,288 news articles published by the *New York Times*'s International section mentioning the word "China." We tokenize the articles by unigrams, remove no stop words, and perform no stemming. The data encodes the binary {0, 1} values (corresponding to appearance/non-appearance) of 86,500 tokens.

Figure 31.3 shows the percentage of explained variance as a function of cardinality. Here we see DSPCA does outperform Thresholded PCA, though not by a big margin. Although we do not have ground truth, Tables 31.2 and 31.3 contain words selected by two algorithms respectively as we increase cardinality. Words selected by DSPCA appear much more meaningful than those chosen by thresholded PCA at the same cardinality.



**Fig. 31.3** Sparse PCA on 1,288 *New York Times* articles mentioning the word “China”

**Table 31.2** 1st PC from DSPCA on 1,288 *New York Times* articles mentioning the word “China” for various values of the eigenvector cardinality  $k$

$k = 2$	$k = 3$	$k = 9$	$k = 14$
United States	American United States	Washington American Administration	International Would Will
		United States	Washington American
		President	Administration
		Obama	United
		Countries	States
		Nations	President
			Obama
			Counties
			Nations
			Policy
			Nuclear

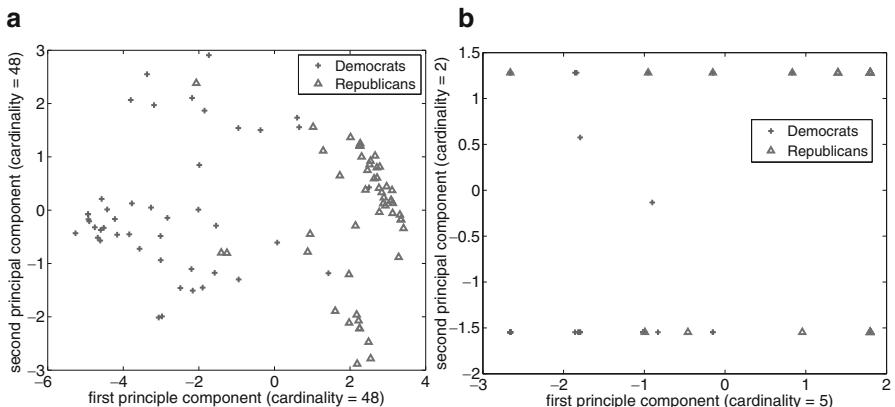
### 31.4.2 Senate Voting Data

In this section, we analyze the voting records of the 109th US Senate (2000–2004). There were 101 senators (one extra senator is due to a replacement during the term) and 48 bills involved. To simplify, the votes are divided into yes (coded as 1) or no (coded as -1), and other votes are coded as 0.

Each senator's voting record can be viewed as a point in a 48-dimensional space. By applying PCA, and projecting each senator's voting record onto a two-dimensional subspace of maximum variance, we can see that senators are almost perfectly separated by partisanship (Fig. 31.4). However, since the principal components involve all the bills, it is hard to tell which bills are most responsible for the explained variance. By applying Sparse PCA to the voting record, we aim to find a few bills that not only divide the senators according to partisanship, but also reveal

**Table 31.3** 1st PC from Thresholded PCA on 1,288 *New York Times* articles mentioning the word “China” for various values of the eigenvector cardinality  $k$

$k = 2$	$k = 3$	$k = 9$	$k = 14$
Even	Even	Even	Would
Like	Like	We	New
	States	Like	Even
		Now	We
		This	Like
		Will	Now
		United	This
		States	Will
		If	United
		United	States
		World	So
		Some	Some
		If	If

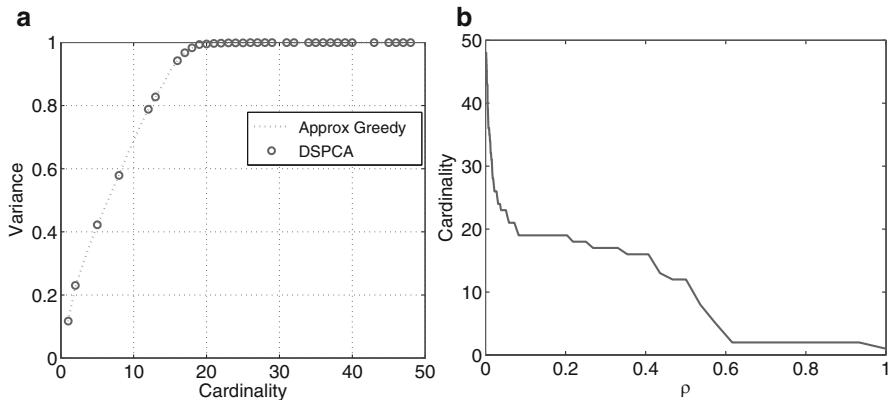


**Fig. 31.4** 109th Senate’s voting record projected onto the top two principal components

which topics are most controversial within the Republican and Democratic parties. Figure 31.4 (right) shows the senators’ voting records, projected onto the first two sparse principal components. We note that in the two-dimensional space senators are still divided by partisanship. In fact, many republican senators perfectly coincide with each other and so are democratic senators. In contrast to Fig. 31.4 (left), the cardinalities associated with the first and second sparse principal components are 5 and 2 respectively, which makes it possible to interpret the coordinates.

Let us examine the bills appearing in the first two sparse principal components. For the first sparse PC, the corresponding bills’ brief description is as follows:

- S. 1932, As Amended; Deficit Reduction Act of 2005.
- S. Con. Res. 83; An original concurrent resolution setting forth the congressional budget for the United States Government for fiscal year 2007 and including the appropriate budgetary levels for fiscal years 2006 and 2008 through 2011.



**Fig. 31.5** Left: Explained variance as a function of cardinality. Right: Cardinality as a function of penalty parameter  $\rho$

- S. 3930, As Amended; Military Commissions Act of 2006.
- S. 403, As Amended; Child Interstate Abortion Notification Act.
- Passage of S. 397, As Amended; Protection of Lawful Commerce in Arms Act.

The brief description for the two bills in the second sparse principal component are:

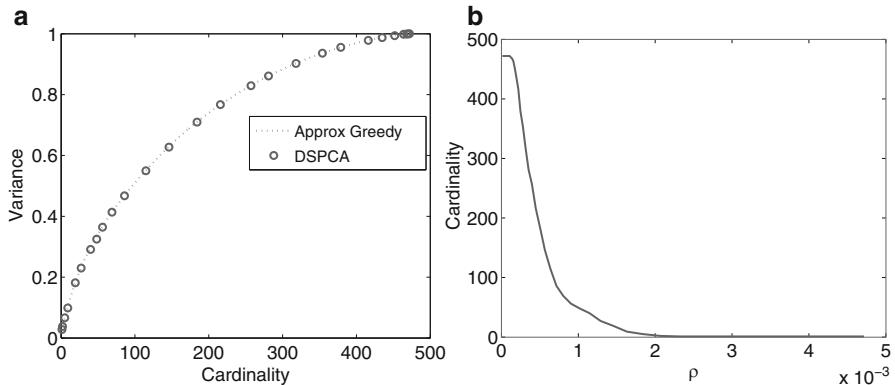
- H. R. 3045; Dominican Republic-Central America-United States Free Trade Agreement Implementation Act.
- S. 1307; Dominican Republic-Central America-United States Free Trade Agreement Implementation Act.

A glance at these bills tells us that the major controversial issues between Democrats and Republicans are topics such as “abortion”, “military”, “budget”, and “free trade”.

Figure 31.5 plots the variance explained by the first sparse principal component divided by that explained by the first PC, as a function of the cardinality of the sparse PC. Figure 31.5 also shows how the cardinality of the first sparse PC varies as the penalty parameter  $\rho$  is changed in the DSPCA code. We can see that when 19 out of 48 variables (bills) are used, sparse PCA almost achieves the same statistical fidelity as standard PCA does.

### 31.4.3 Stock Market Data

In this section, we investigate the historical prices of S&P500 stocks over 5 years, from June 1st, 2005, through June 1st, 2010. By taking out the stocks with less than 5 years of history, we end up with 472 stocks, each having daily closing prices over



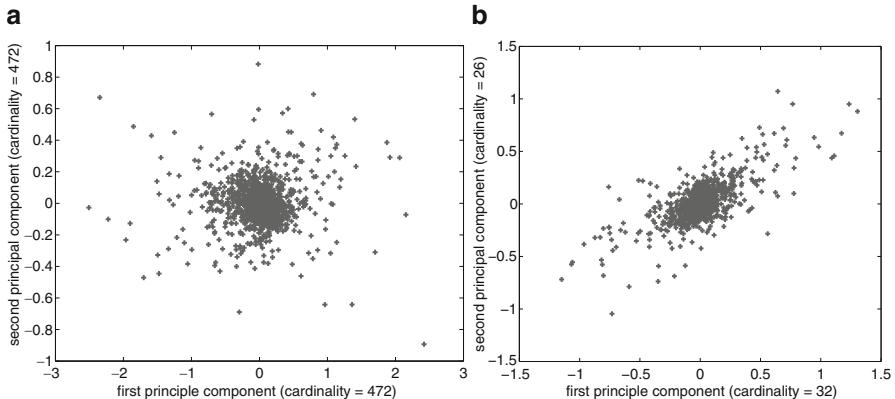
**Fig. 31.6** *Left:* Explained variance as a function of cardinality. *Right:* Cardinality as a function of penalty parameter  $\rho$

1,259 trading days. The prices are first adjusted for dividends and splits and then used to calculate daily log returns. Each day's return can be represented as a point in  $\mathbf{R}^{472}$ .

Figure 31.6 shows the explained variance as a function of 1st PC's cardinality. It seems hard to say that the 1st PC is sparse, since there is no natural “kink” in that curve. That is, we need almost 300 out of the total 472 stocks to explain at least 90% of the variance explained by the 1st PC from PCA. However, when we inspect the sparse PCs with increasing cardinalities, we note that initially only stocks from the “Financials” sector come to play and later until, at cardinality 32, do we see companies from other sectors appearing in the 1st sparse PC. So we take the first sparse PC with cardinality equal to 32. Then we solve for the 2nd sparse PC, and using the same guideline to arrive at a cardinality of 26.

Figure 31.7 show the stock returns projected onto the 2-dimensional subspaces spanned by the top two PCs and top two sparse PCs, respectively. Comparing these two plots, we observe two interesting phenomena:

- Although the top two principal components from PCA explain more variance (as seen from the larger range of the axes in the left over the right panel), the two sparse principal components from DSPCA involve only 58 out of 472 stocks (32 on the first PC and another distinct 26 on the second). Furthermore, 31 of the 32 stocks in the first sparse PC are all from the sector “Financials”, and that almost all 26 stocks in the second sparse PC come from “Energy” and “Materials” except 2 from “Financials” and 1 from “Information Technology”. Considering that there are ten sectors in total, this is quite interesting as Sparse PCA is able to identify the right groups (industry factors) that explains most of the variance. Our data covers June 2005 through June 2010 where a severe financial crisis took place, and the key role of the Financial sector is revealed purely through our sparse PCA analysis.

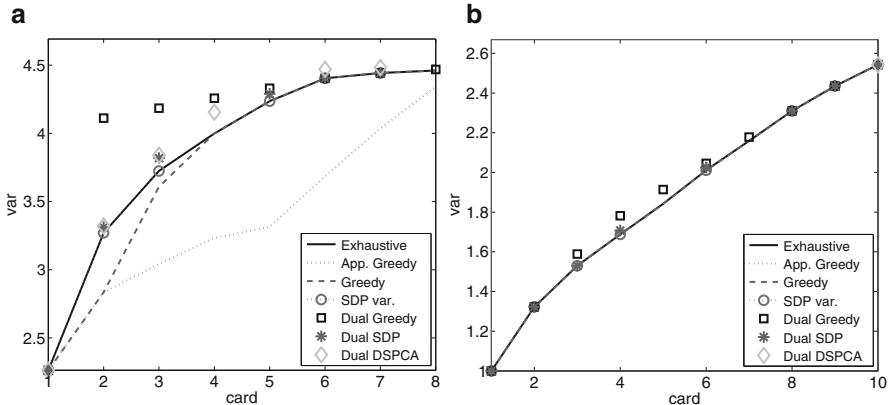


**Fig. 31.7** S&P500 daily returns projected onto the top two principal components. For PCA (*left*) and sparse PCA (*right*)

- In Fig. 31.6, the projected data appears symmetrically distributed around its center. In contrast, In Fig. 31.7 (right), we observe a definite orientation. Since the horizontal axis (first PC) corresponds to “Financials” and the vertical one to “Energy” and “Materials”, the sparse PCA analysis tells us that these two sectors are positively correlated.

### 31.4.4 Random Matrices

Sparse eigenvalues of random matrices play a central role in characterizing the performance of  $\ell_1$  decoders in compressed sensing applications. Testing the Restricted Isometry Property (RIP) in [6] amounts to bounding the maximum and minimum eigenvalues of a Gram matrix. Here, we compute the upper and lower bounds on sparse eigenvalues produced using various algorithms. We pick the data matrix to be small enough so that computing sparse eigenvalues by exhaustive search is numerically feasible. In Fig. 31.8, we plot the maximum sparse eigenvalue versus cardinality, obtained using exhaustive search (solid line), the approximate greedy (dotted line) and fully greedy (dashed line) algorithms. We also plot the upper bounds obtained by minimizing the gap of a rank one solution (squares), by solving the semidefinite relaxation in Sect. 31.2.2 explicitly (stars) and by solving the DSPCA dual (diamonds). On the left, we use a matrix  $\Sigma = F^T F$  with  $F$  Gaussian. On the right,  $\Sigma = uu^T / \|u\|^2 + 2V^T V$ , where  $u_i = 1/i$ ,  $i = 1, \dots, n$  and  $V$  is matrix with coefficients uniformly distributed in  $[0, 1]$ . Almost all algorithms are provably optimal in the noisy rank one case (as well as in many example arising from “natural data”), while Gaussian random matrices are harder. Note however, that the duality gap between the semidefinite relaxations and the optimal solution is very small in both cases, while our bounds based on greedy solutions are not as good. Overall,



**Fig. 31.8** Upper and lower bound on sparse maximum eigenvalues. We plot the maximum sparse eigenvalue versus cardinality, obtained using exhaustive search (*solid line*), the approximate greedy (*dotted line*) and fully greedy (*dashed line*) algorithms. We also plot the upper bounds obtained by minimizing the gap of a rank one solution (squares), by solving the  $\ell_0$  semidefinite relaxation explicitly (stars) and by solving the DSPCA dual  $\ell_1$  relaxation (diamonds). *Left:* On a matrix  $F^T F$  with  $F$  Gaussian. *Right:* On a sparse rank one plus noise matrix

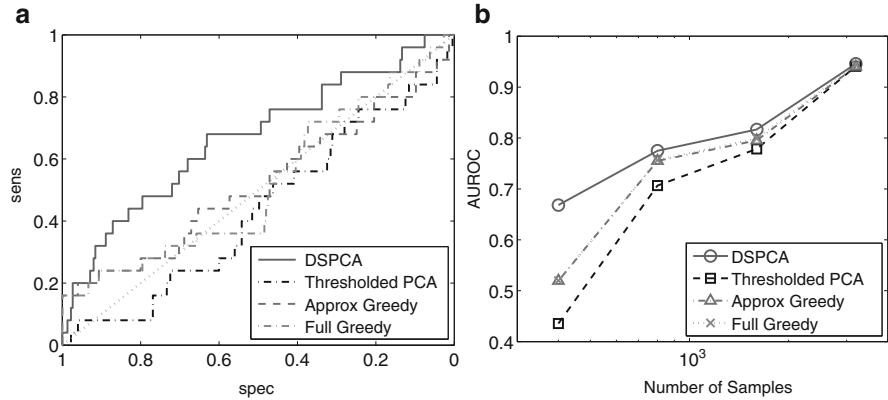
while all algorithms seem to behave similarly on “natural” or easy data sets, only numerically expensive relaxations produce good bounds on the random matrices used in compressed sensing applications.

### 31.4.5 Statistical Consistency vs. Computational Complexity

As we hinted above, very simple methods such as thresholding or greedy algorithms often perform well enough on simple data sets, while obtaining good statistical fidelity on more complex (or random) data sets requires more complex algorithms. This is perfectly illustrated by the results in [2] on a spiked covariance model. To summarize these results, suppose that the sample covariance matrix  $\hat{\Sigma} \in \mathbf{S}_n$  is a noisy estimate of the true population covariance  $\Sigma \in \mathbf{S}_n$  with  $\hat{\Sigma} = \Sigma + \Delta$  where  $\Delta$  is a noise matrix, suppose also that the leading eigenvector of the true covariance is sparse with cardinality  $k$ . Under some assumptions on the noise component  $\Delta$ , [2] show that when the ambient dimension  $n$ , the number of observations  $m$  and the number  $k$  of nonzero components in the leading eigenvector all scale to infinity, and when the ratio

$$\theta_{\text{thres}} = \frac{m}{k^2 \log(n-k)}$$

is above some critical value, then simply thresholding the diagonal of the sample covariance matrix will recover the exact support of the leading eigenvector of  $\Sigma$  with probability tending to one. On the other hand, simple thresholding fails with probability one when this ratio is below a certain value. Furthermore, [2] show that when



**Fig. 31.9** Left: ROC curves when recovering the support of  $u$  in the spiked model using thresholding, approximate and exact greedy algorithms and the semidefinite relaxation (DSPCA) in Sect. 31.2.1 in the spiked model when  $n = 250$ ,  $m = 400$  and  $k = 25$ . Right: Area Under ROC (AUROC) versus number of samples  $m$

$$\theta_{\text{sdp}} = \frac{m}{k \log(n - k)}$$

is above some critical value, the solution of the semidefinite relaxation in Sect. 31.2.1 (if tight) will recover the exact support of the leading eigenvector of  $\Sigma$  with probability tending to one. On the other hand, the semidefinite relaxation fails with probability one when this ratio is below a certain value. They also show that the semidefinite programming relaxation in Sect. 31.2.1 is statistically optimal, meaning that no other method (even combinatorial ones) can recover the true support using fewer samples (up to a constant factor). This result clearly illustrates a tradeoff between statistical fidelity on one side and computational complexity on the other. In the spiked model, the semidefinite relaxation requires  $O(1/k)$  fewer samples than simply thresholding the diagonal to recover the true support of the leading eigenvector of  $\Sigma$ , but its complexity is much higher than that of the thresholding strategy.

We can further illustrate this behavior on a simple numerical example. Suppose we are given a sample covariance  $\hat{\Sigma} \in \mathbf{S}_n$  coming from a “spiked” model of covariance similar to that in [2], with

$$\hat{\Sigma} = uu^T + VV^T / \sqrt{m}$$

where  $u \in \mathbf{R}^n$  is the true sparse leading eigenvector, with  $\text{Card}(u) = k$ ,  $V \in \mathbf{R}^{n \times m}$  is a noise matrix with  $V_{ij} \sim \mathcal{N}(0, 1)$  and  $m$  is the number of observations. We compare the performance of the simple thresholding method (on the leading eigenvector of regular PCA here) with that of the semidefinite relaxation when recovering the support of  $u$  for various values of the number of samples. Our point here is that, while variance versus cardinality is a direct way of comparing the performance

of sparse PCA algorithms, accurate recovery of the support is often a far more important objective. Many methods produce similar variance levels given a limited budget of nonzero components, but their performance in recovering the true support is often markedly different.

In Fig. 31.9 on the left we compare ROC curves when recovering the support of  $u$  in the spiked model above using thresholded PCA, the approximate and full greedy algorithms in [7] and semidefinite relaxation (DSPCA). On the right, we plot Area Under ROC as the number of samples increase. As expected, we observe that the semidefinite relaxation performs much better when only a limited number of observations are available ( $m$  small).

## 31.5 Conclusion

We have reviewed here several techniques for approximating the solution to the *single factor* sparse PCA problem. While the algorithms presented here perform quite well, several key questions remain open at this point.

First, outside of the (locally) convergent algorithm in [14], very few methods handle the problem of simultaneously finding several leading sparse principal components.

Also, as the examples of Sect. 31.4 illustrate, most methods (even extremely simple ones) perform well enough on easy, “natural” data sets while only the most expensive semidefinite relaxations seem to produce good bounds on the random matrices used in compressed sensing applications, or when only a few samples are available for example. Characterizing what makes “natural” data sets easier than random ones remains an open problem at this point. It is also not clear yet how to extend the statistical optimality statements of [2] to broader (e.g. deterministic) classes of matrices.

Finally, the question of approximation bounds à la MAXCUT for the relaxations detailed here is largely open. Basic performance bounds are discussed in [3, 8] but they can certainly be tightened.

**Acknowledgements** The authors gratefully acknowledge partial support from NSF grants SES-0835550 (CDI), CMMI-0844795 (CAREER), CMMI-0968842, a Peek junior faculty fellowship, a Howard B. Wentz Jr. award and a gift from Google.

## References

1. Alizadeh, F.: Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization* **5**, 13–51 (1995)
2. Amini, A.A., Wainwright, M.: High-dimensional analysis of semidefinite relaxations for sparse principal components. *The Annals of Statistics* **37**(5B), 2877–2921 (2009)
3. Bach, F., Ahipasaoglu, S.D., d’Aspremont, A.: Convex relaxations for subset selection. working paper, 2010.

4. Boyd, S., Vandenberghe, L.: *Convex Optimization*. Cambridge University Press (2004)
5. Cadima, J., Jolliffe, I.T.: Loadings and correlations in the interpretation of principal components. *Journal of Applied Statistics* **22**, 203–214 (1995)
6. Candès, E.J., Tao, T.: Decoding by linear programming. *IEEE Transactions on Information Theory* **51**(12), 4203–4215, 2005.
7. d'Aspremont, A., Bach, F., El Ghaoui, L.: Optimal solutions for sparse principal component analysis. *Journal of Machine Learning Research* **9**, 1269–1294 (2008)
8. d'Aspremont, A., El Ghaoui, L.: Testing the nullspace property using semidefinite programming. *Mathematical Programming, Series B*, **127**(1), 123–144 (2011)
9. d'Aspremont, A., El Ghaoui, L., Jordan, M.I., Lanckriet, G.R.G.: A direct formulation for sparse PCA using semidefinite programming. *SIAM Review* **49**(3), 434–448 (2007)
10. Donoho D.L., Tanner, J.: Sparse nonnegative solutions of underdetermined linear equations by linear programming. *Proceedings of the National Academy of Sciences* **102**(27), 9446–9451 (2005)
11. Fazel, M., Hindi, H., Boyd, S.: A rank minimization heuristic with application to minimum order system approximation. *Proceedings American Control Conference* **6**, 4734–4739 (2001)
12. Golub, G.H., Van Loan, C.F.: *Matrix computation*. North Oxford Academic (1990)
13. Horn, R.A., Johnson, C.R.: *Matrix Analysis*. Cambridge University Press (1985)
14. Journée, M., Nesterov, Y., Richtárik, P., Sepulchre, R.: Generalized power method for sparse principal component analysis. arXiv:0811.4724 (2008)
15. Jolliffe, I.T.: Rotation of principal components: choice of normalization constraints. *Journal of Applied Statistics* **22**, 29–35 (1995)
16. Jolliffe, I.T., Trendafilov, N.T., Uddin, M.: A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics* **12**, 531–547 (2003)
17. Kaiser, H.F.: The varimax criterion for analytic rotation in factor analysis. *Psychometrika* **23**(3), 187–200 (1958)
18. Lemaréchal, C., Oustry, F.: Semidefinite relaxations and Lagrangian duality with application to combinatorial optimization. INRIA, Rapport de recherche, 3710 (1999)
19. Lovász, L., Schrijver, A.: Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization* **1**(2), 166–190 (1991)
20. Mackey, L.: Deflation methods for sparse pca. *Advances in Neural Information Processing Systems* **21**, 1017–1024 (2009)
21. Moler, C., Van Loan, C.: Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review* **45**(1), 3–49 (2003)
22. Moghaddam, B., Weiss, Y., Avidan, S.: Generalized spectral bounds for sparse LDA. In: *International Conference on Machine Learning* (2006)
23. Moghaddam, B., Weiss, Y., Avidan, S.: Spectral bounds for sparse PCA: exact and greedy algorithms. *Advances in Neural Information Processing Systems* **18**, 915–922 (2006)
24. Moghaddam, B., Weiss, Y., Avidan, S.: Fast Pixel/Part Selection with Sparse Eigenvectors. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8 (2007)
25. Natarajan, B.K.: Sparse approximate solutions to linear systems. *SIAM J. Comput.* **24**(2), 227–234 (1995)
26. Nesterov, Y.: A method of solving a convex programming problem with convergence rate  $O(1/k^2)$ . *Soviet Mathematics Doklady* **27**(2), 372–376 (1983)
27. Nesterov, Y.: *Introductory Lectures on Convex Optimization*. Springer (2003)
28. Nesterov, Y.: Smoothing technique and its applications in semidefinite optimization. *Mathematical Programming* **110**(2), 245–259 (2007)
29. Neuhaus, J.O., Wrigley, C.: The quartimax method: an analytical approach to orthogonal simple structure. *British Journal of Statistical Psychology* **7**, 81–91 (1954)
30. Saad, Y.: *Numerical methods for large eigenvalue problems*. Manchester Univ Press (1992)
31. Sriperumbudur, B.K., Torres, D.A., Lanckriet, G.R.G.: Sparse eigen methods by DC programming. In: *Proceedings of the 24th international conference on Machine learning*, Corvallis, USA 20–24 2007

32. Tibshirani, R.: Regression shrinkage and selection via the LASSO. *Journal of the Royal statistical society, series B* **58**(1), 267–288 (1996)
33. Zou, H., Hastie, T., Tibshirani, R.: Sparse Principal Component Analysis. *Journal of Computational & Graphical Statistics* **15**(2), 265–286 (2006)
34. Zhang, Z., Zha, H., Simon, H.: Low rank approximations with sparse factors I: basic algorithms and error analysis. *SIAM journal on matrix analysis and its applications* **23**(3), 706–727 (2002)

# Index

## A

Adjacency matrix  
bandwidth problem, 814  
canonical circuit, 804  
definition, 805  
and  $\text{OPT}_{\text{MC}}$ , 814  
relaxation coincides, 813  
SDP relaxation, 811  
Adjoint, 605, 609, 882  
Affine variety, 62, 64, 274  
AHO direction, 737  
AHO method, 331, 332  
Ai-Zhang direction, 453  
Albert algebra, 320  
Algebra  
element, 416  
functions, 408, 417, 418  
Algebraic certificates  
description, 396  
Nullstellensatz, 399  
Positivstellensätze, 396–398  
quotient algebras, 398  
tracial Positivstellensatz, 400–401  
Algebraic degree derivation  
algebra geometric methods  
Bertini’s theorem, 66  
Bezout’s theorem, 66–67  
degree of variety, 66  
intersection theory, 66  
complex projective setup  
genericity conditions, 65  
homogeneous coordinate functions, 63  
polynomial equations, 63  
projective variety, 63  
Zariski closure, 64

degree formulas  
polynomial optimization problem, 68–69  
QCQP, 70  
SDP, 71–72  
genericity assumptions and KKT  
constraint functions, 65  
upper bound, 65–66  
Algebraic sets, convex hulls  
arbitrary ideals, 114  
convergence and exactness  
2-level polytopes, 132  
3-level polytopes, 132–133  
 $\text{TH}_1$  and  $\text{TH}_2$ , 133  
theta bodies, 131  
description, 113–114  
max-cut problem, 134–135  
permutation groups, 135–136  
polynomial ideals, theta bodies, 115–121  
real radical ideals, 114  
theta bodies computation, 121–128  
Algebraic software  
NCAlgebra Under Mathematica, 401  
NCSOStools Under Matlab, 402  
Algorithm IPC, 752  
Almost periodic function, 424  
Alternating direction method, 573  
Alternating projection method, 571  
Analytic center and cutting-surface method, 480–483  
Analytic centers  
dual feasible region, 485  
IPMs, 483  
Angular distance, 250–251, 257, 259, 260  
Approximate greedy, 927

- Approximation algorithms  
 coloring 3-colorable graphs  
 chromatic number, 153  
 Minimum Vertex Cover, 154  
 NP-complete problem, 152  
 SDP solution, 154
- hypergraph independent set  
 Chromatic Number, 155  
 MINOPT, 155–156  
 mixed level- $t$  relaxation, 155
- hyperplane rounding, 148
- LP and SDP relaxation, 148
- max-cut and hyperplane rounding, 148–149
- Sherali–Adams and Lasserre relaxations,  
 Knapsack  
 integral hull, 158–159  
 parametrized feasibility LP, 158  
 standard LP, 157
- sparsest cut and metric embedding  
 ARV-round, 150–152  
 local SDP relaxation, 149
- Archimedean property, 399
- Association schemes  
 description, 174  
 eigenvalues, 175–176  
 sources, 174–175
- Asymptotically feasible, 351
- Augmented Lagrangian  
 algorithm, 681  
 alternating direction method, 573  
 approaches, 579  
 dual, primal proximal algorithm, 583–584
- Augmented Lagrangians method, RBR  
 general linear constraints, 549–550  
 positive definiteness, 550  
 quadratic penalty function, 550
- Automorphism group (of a symmetric cone), 322
- Automorphism group (of an algebra), 136
- Automorphism groups, 798–800, 803, 804, 813
- B**
- Bandwidth of a graph, 814
- Bandwidth of the labeling, 813–814
- Bandwidth problem, 813
- Barrier function, 474
- Barrier method, 683
- Benchmarks  
 MCNC, 873  
 PENSMP, 674  
 SDPs, 705
- Bilinear matrix inequality, 776
- Binary code, 194, 242, 248
- Bernstein theorem, 423
- Bertini’s theorem, 66
- Bessis–Moussa–Villani (BMV), 378, 400
- Betweenness  
 based LP formulation, 859–861  
 polytope, 858
- Bezout’s theorem, 66–67
- Biclique completion problem  
 CP model  
 domain filtering algorithms, 660  
 one-shore induced quasi-biclique constraint, 659–660
- evaluation, hybrid approach  
 optimal solution matrix, 662  
 search heuristic, 662  
 variants, 663
- multicast services  
 clustering, 658–659  
 vs. unicast services, 657–658
- problem description, 659
- SDP model, 660–662
- BiqMac algorithm  
 description, 839–840, 843–844  
 extended version, 827
- Biswas–Ye relaxation, 903
- Block codes  
 generalizations, 248  
 Hamming space and distance, 242  
 hierarchies and  $k$ -tuples, 245–247  
 Lovász and Delsarte’s linear programming bound, 242–244  
 triples, 244–245
- Block coordinate descent (BCD) method  
 algorithmic strategy, 534  
 complexity results, 535  
 convergence properties, 535  
 convergence results, 546–548  
 description, 534  
 Gauss–Southwell rule, 534  
 matrix completion, 543–545  
 max-cut SDP relaxation, 540–543  
 notation and organization, 537
- RBR method  
 augmented Lagrangian method, 549–552  
 matrix completion, 553–554  
 max-cut SDP, 552  
 prototype, 538–540
- Schur complement, 538
- sparse inverse covariance estimation, 545–546

- Block diagonalization  
advantage, 225–226  
\*–isomorphism, 226–227  
matrix \*–algebra  
algorithmic aspects, 233  
description, 224–225  
group representations, 233  
\*–homomorphisms and, 231  
\*–subalgebra, 231–232  
unitary matrix, 232–233  
matrix multiplication, 223
- Block-diagonal structure, 482, 507, 677, 825
- Block structure, 226, 232, 690, 691, 710, 718, 719–721, 723, 732
- BMV. *See* Bessis–Moussa–Villani
- Bochner Theorem, 425
- Border base, 26, 27, 35–38
- Borel measure, 12–14, 41, 42, 91, 235, 274, 277, 281, 400, 410, 411, 423
- Bose–Mesner algebra, 229
- B-subdifferential, 355
- Bundle method  
ConicBundle, 682–683  
nonsmooth optimization, 472  
polyhedral/spectral, 483
- C**
- Canonical basis, 14, 223, 227, 229, 234, 243, 276
- Canonical circuit, 804
- Cell arrays, 718–726, 731, 732, 743, 744
- Center, 231
- Centered subspace, 884
- Centrality measure, 330, 332
- Central path  
IPMs, 485, 489  
Newton system, 475  
primal-dual, 474
- Certificate of infeasibility, 397, 723, 724
- Certificate of positivity, 415, 421
- C-function, 355–357, 359, 360, 362
- Characteristic polynomial, 306–309, 311, 312, 325
- Chen–Harker–Kanzow–Smale (CHKS)  
smoothing function, 353, 358, 366
- Chen–Mangasarian smoothing functions, 361
- Cholesky factor  
and chordal graph, sparse, 508–510  
CPLEX, 684  
CSDP, 677  
LOQO, 680–681
- Chordal graph  
Cholesky factorization, 508–510  
minimal vertex separator, 889
- Chordal extension, 509–511, 514
- Chromatic number, 825
- Clarke’s generalized Jacobian, 355, 557
- Clifford algebra, 305
- Clique, 480
- Clique inequalities, 860
- Codes. *See* Conic optimization software
- Cohen–Macaulay property, 261
- Coherent configuration, 229
- Coloring, 152–154, 165, 264, 824–826, 845
- Column generation  
ACCPM approach, 482  
IPMs, 478
- Combinatorial moment matrix, 116, 245
- Combinatorial optimization problems  
bandwidth problem  
labeling, 813–814  
Laplacian eigenvalue, 815–816  
lower bounds, 814  
min-cut problem, 814  
 $\text{OPT}_{\text{MC}}$  bounds, 814–815  
biclique completion problem, 657–663  
 $k$ -equipartition problem ( $k$ -GP), 804–813  
notation, 795–796  
quadratic assignment problem (QAP), 797–803, 814  
traveling salesman problem (TSP), 803–804
- Combinatorial problems relaxations  
association schemes  
eigenvalues, 175–176  
sources, 174–175  
coherent configurations, 173  
lift-and-project techniques, 171  
maximum bisection problem, 172  
preliminaries and notation, 173–174  
semidefinite programming, 176–197
- Commutant, 229
- Commutative algebra, 232, 239, 303, 329
- Commutative class of directions, 332, 349, 351
- Commutative class of scalings, 348
- Commutator, 630
- Companion matrix, 34, 35
- Complementarity and barriers  
definition, 299  
description, 299  
interior point methods  
definition, 301  
function, 301  
Lagrangian function, 301–302  
Newton’s method, 303  
nonnegative orthant, 302  
positive semidefinite cone, 302–303

- Complementarity and barriers (*cont.*)  
     second order cone, 302  
     left and right multiplication, 303  
     proper cones, 300–301
- Complementarity problems over symmetric cones  
     definition, 339  
     description, 339  
     duality theorem, 341–342  
     Euclidean Jordan algebra, 342–344  
     implicit SCCP, 339–340  
     interior-point methods, SCCP (*see*  
         Interior-point methods)  
     merit/smoothing function methods (*see*  
         Merit/smoothing function methods)  
     monotone SCCP, 340–341  
     SCCP properties (*see* SCCP)  
     standard SCCP, 339–340  
     vertical SCCP, 340
- Complementarity set, 299, 301, 335
- Complementary cone of  $L$ , 369
- Complementary face, 369
- Complete bipartite graph, 172, 174, 175, 180,  
     194, 220, 248, 659
- Complete intersection, 67, 69, 70
- Completely positive, 201, 203, 214, 215, 381,  
     386, 387
- Completely positive map, 381, 386
- Completely positive matrix, 214
- Completely positive program, 201, 215, 216
- Completely positive programming, 201, 215,  
     216
- Complete positivity. *See* Linear matrix  
     inequality (LMI)
- Complete system of orthogonal idempotents,  
     313, 314, 343
- Completion method  
     SDPA-C (*see* SDPA family developments)  
     SDPARA-C (*see* SDPA family  
         developments)
- Complexity  
     Hessian assembling  
         block diagonal cases, 766–768  
         full matrices, 766  
     Newton system solution  
         Cholesky method, 768  
         Hessian formula, 769  
         iterative algorithms, 768–769
- Complex root, 26–29, 31, 33–38, 51–52
- Computational complexity. *See* Statistical  
     consistency
- Condition number, 333, 785–787
- Cone of critical directions, 462, 464, 467,  
     757
- Cone optimization problem, 671
- Cone of squares  
     definition, 320  
     symmetric, 322
- Conic optimization  
     positive semidefinite matrices  
         Cholesky decomposition, 9  
         definition, 6–8  
         Schur complement theorem, 9  
         spectral theorem, 8
- semidefinite programming  
     combinatorial problems, 6  
     computational complexity, 11  
     duality, 9–10  
     Laplacian matrix, 4  
     linear matrix inequality (LMI), 2  
     linear programming (LP) problems, 1  
     max-cut problem, 2–3, 134, 821  
     relaxation, 5–6
- Conic optimization software  
     codes  
         ConicBundle, 682–683  
         CPLEX, 684  
         CSDP, 677  
         DSDP, 677–678  
         groups and input formats, 675  
         LOQO, 680–681  
         MOSEK, 680  
         SDPA, 675–676, 687, 693  
         SDPLR, 681–682  
         SDPT3, 678–679  
         SeDuMi, 676  
         SMCP, 683  
     computing errors  
         evaluation, 674  
         optimal solutions, 673–674  
     problems  
         definition, 671–672  
         feasible set, 672–673  
         rotated quadratic, 673
- Conic projection problem, 567–568
- Consistency  
     domain, 640–641  
     statistical vs. computational complexity,  
         936–938
- Constraint programming (CP)  
     combinatorial problem, 635  
     domain filtering algorithms, 636  
     global constraint, 638  
     integration, semidefinite relaxations, 636  
     model, 659–660  
     modeling  
         Boolean functions, 637–638  
         global constraints, 638

- vs. ILP, 637
  - set variables, 638–639
- optimization constraints
  - assignment problem, 642–643
  - cost-based domain filtering, 642
  - description, 641–642
  - weight function, 642
- search strategies
  - best-bound-first strategy, 644
  - branching decision, 643
  - limited discrepancy, 644, 645
  - local branching, 644
  - variable and value selection heuristics, 643–644
- semidefinite relaxations
  - Boolean satisfiability problem, 648–649
  - construction, 646
  - optimization constraint, 647
  - search heuristic, 647–648
  - weighted CSPs, 649
- solver and systems, 635
- solving
  - constraint propagation, 639–640
  - domain consistency, 640–641
  - search process, 639
- Constraint qualification, 10, 212, 458–461, 463–465
- Constraint satisfaction problems (CSPs)
  - integrality gaps, proofs, 162
  - Lasserre hierarchies, 163–164
  - Lovász–Schrijver hierarchies, 163
  - weighted, 649
- Convergence theory, 757–759
- Convex cone, 1, 201, 203, 214, 235, 299, 345, 369, 396, 409, 565, 577–578, 828, 857, 882, 883, 886
- Convex hulls. *See also* Semidefinite representation, convex sets and hulls
  - extraneous variables, 206–208
  - implied constraints, 205–206
  - localization and SD $\sigma$  construction
    - defining polynomials, 106–107
    - homogeneous, 106
    - Minkowski sum, 104
    - nonempty convex sets, 105
    - SDP representation, 107–108
    - unboundedness, 105–106
    - quadratic constraints, 208–209
- Convexity
  - rigid, 83–84
  - SOS, 95
- Convex optimization, 471, 581
- Convex polynomials
  - NCAlgebra command, 392–393
  - symmetric polynomial, 392
- Convex quadratic programming bound, 796
- Convex relaxations
  - approximation algorithms, 147–159
  - FRAC/OPT ratio, 140
  - FRAC/ROUND, 140
  - hierarchies, 140–146
  - maximization problem, 140
  - NP-hard optimization problems, 139
- Convex semi-algebraic set, 577–578
  - bounded nonconvex, 79
  - boundary, 89
  - compact, 90
  - conditions, SDP representability, 90
  - convex set separation theorem, 89
  - nonempty interior, 78
  - problem and conditions, 88–89
- Convex sets. *See also* Semidefinite representation, convex sets and hulls
  - SOS (*see* Sum-of-squares (SOS))
  - strictly
    - concave defining polynomials, 98–102
    - quasi-concave defining polynomials, 102–104
- Copositive programming
  - applications, 215–216
  - completely positive, 201
  - convex hull, 203–209
  - doubly nonnegative matrices, 214
  - duality, 211–212
  - results, 212–213
  - symmetric matrix, 201
- Corrector-predictor approach, 475
- Correlation matrix
  - nearest correlation matrix problem, 578–579
  - projection, 566
- Correlative sparsity, 510, 695
- Correlative sparsity pattern graph, 510
- Correlative sparsity pattern matrix, 508
- Counting measure, 235, 251
- Courant function, 425
- CP. *See* Constraint programming
- Critical directions, cone, 462–463, 757
- Critical locus, 63, 65, 66, 69, 70, 73
- Cross Commutative property, 363
- Crossing numbers, 248–250
- CSPs. *See* Constraint satisfaction problems
- Cut generation, 478–479, 556
- Cut inequalities, 481–482
- Cut polytope, 479, 480
- Cut separation, 161, 858

- Cutting plane algorithm  
 analytic-center and cutting-surface method, 480–483  
 primal-dual IPMs, 483–484  
 SDP relaxation and cutting planes, 479–480
- Cutting-plane method, 480–483
- Cutting-plane separation, 838
- Cutting-surface method, 480–483
- Cycle inequalities, 480, 838
- D**
- Decomposition, 27, 62, 87, 214, 238, 261, 334, 384, 640, 641, 906, 915
- Degeneracy, 477
- Degree of a Jordan algebra, 314, 316
- Degree of an element, 71
- Degree of variety, 69
- Delay system, 421, 429, 432
- Delsarte’s linear programming bound, 242–244, 257, 259
- Dense columns, 676, 684, 739–743, 745
- Dense SDP relaxation, 503–506
- Derivations, 3, 63–75, 215, 334, 443, 516, 525
- Determinantal varieties, 67–69, 71
- Dihedral group, 248, 804
- Dimension-free matrix  
 algebraic certificates  
 description, 396  
 Nullstellensatz, 399  
 Positivstellensätze, 396–398  
 quotient algebras, 398  
 tracial Positivstellensatz, 400–401
- algebraic software  
 NCAlgebra Under Mathematica, 401  
 NCSOStools Under Matlab, 402
- complete positivity and LMI domination, 383–388
- convex polynomials, 392–395
- convex semi-algebraic sets, 390
- inequality, 377–378
- LMI  
 domination problem, 381  
 representation, 389
- matrix convex, 382
- monic linear pencil, 383
- motivation, 391–392
- non-commutative basic open semi-algebraic sets, 390
- non-commutative polynomials  
 indeterminates, 379–380  
 involution, 379  
 LMIs, 380–381
- matrix-valued polynomials, 380  
 polynomial  $p$ , 379  
 Positivstellensatz, 383
- Direct sum, 34, 45, 226–228, 231, 232, 236, 238, 315, 318, 325, 329, 389
- Discrete topology, 235
- Distance-regular graph, 174, 175, 178, 179
- Distance to infeasibility, 746
- Donath-Hoffman bound, 806
- D-optimal experimental design, 732–733
- Doubly nonnegative, 214
- Dual cone, 7, 201, 206, 211, 214, 299, 321, 882
- Dual degenerate, 896
- Duality, 211–212, 473
- Duality gap, 475, 477
- Dual projection algorithm  
 description, 584  
 inner restarting, 585  
 interior-point methods, 586  
 memory, 585  
 outer stopping test, 586  
 overall stopping test, 586  
 variable and outer iterates, 585
- Duals  
 and Positivstellensatz, 616–617  
 SOS decomposition, 620
- Dual-scaling algorithm, DSDP, 841
- E**
- EDM. *See* Euclidean distance matrices
- EDMC. *See* Euclidean distance matrix completion
- Eigenvalue decomposition, 446, 459, 536, 737, 906
- Eigenvalue method, 33–36
- Eigenvalue optimization, 402
- Eigenvalue optimization problem, 482
- Eigenvalues  
 characteristics, 307  
 linear function, 307  
 $M_r, o$ , 308  
 product, 308  
 spin factor algebra, 308  
 theory, 306
- Elementary optimality conditions, nonlinear SDPs  
 applications, areas, 455  
 first order conditions  
 constraint qualification, NLSDP, 460–461  
 KKT conditions, 461  
 linearized subproblems, 457  
 NLSDPs and NLPs, 455–456

- second order conditions
  - critical directions, cone, 462–463, 757
  - example, 464
  - necessary and sufficient conditions, 465–469
  - qualification, 463–464
  - symmetric bilinear mapping, 461–462
- tangential and linearized cones
  - constraint qualifications, NLP, 458–460
  - feasible set, 457–458
- Elementary semi-algebraic functions, 414–417
- Elliptope, 479–480, 839
- Embedding dimension, 882, 886–889, 892, 895, 896, 897
- Error bound, 355, 535
- Error computation, 675–684
- Error correcting code, 220, 242, 251
- Euclidean distance matrices (EDM)
  - distance geometry, 879–880
  - graph realization and rigidity, 880
  - linear maps, 883–886
  - molecular conformation, 881
  - Schoenberg theorem, 886
  - sensor network localization, 880–881, 898–909
- Euclidean distance matrix completion (EDMC)
  - problem
  - chordal, 888–889
  - description, 887
  - facial reduction, 896–898
  - graph realization, 889–890
  - low-dimensional, 888
  - NP-hard, 891
  - rank minimization heuristics
    - maximum trace, 893–894
    - nuclear norm, 895
  - SDP relaxation, 891–893
- Euclidean Jordan algebra, 341–345, 360, 362, 368, 459
- Euclidean space, 409, 413, 425
- Exact penalty function method, 833
- Explained variance, 928–931, 933, 934
- Exposed face, 882
  
- F**
- Face (of a closed convex cone), 369, 565
- Face (of a polytope), 827, 852, 854–858, 861
- Facially exposed cone, 882
- Facility layout and max-cut, 491–493
- Feasibility problem
  - algorithms, 481
  - conic optimization, 482
  - cutting-plane method, 494
- Feasible interior point, 473–475, 487
- First-order methods
  - complexity, 925
  - Euclidean projection, 924–925
  - gradient mapping, 923–924
  - stopping criterion, 925
- First-order optimality conditions, 474
- Fischer-Burmeister function, 357, 359, 360, 368
- Floorplanning. *See* Very large scale integration (VLSI) floorplanning
- Formally real algebra, 305
- Formally real Jordan algebra, 297–335
- Formally Real Jordan algebras
  - complementarity and barriers (*see* Complementarity and barriers)
  - cone of squares and automorphisms, 320–323
  - nondegeneracy and interior point methods, 325–334
- Peirce decomposition, 315–320
- primal-dual interior point methods, 297–298
- properties
  - associative, 303
  - commutative, 303
  - commutator operation, 303
  - Jordan identity, 304–305
  - polarized form, 306
  - polynomials, eigenvalues and inverse elements, 306–309
  - quadratic representation, 309–310
  - spin factor/quadratic algebra, 305
  - symmetric and Hermitian matrices, 305
- quantum physics aspects, 298
- real eigenvalues, 311
- role, 297
- spectral functions, 323–325
- spectral properties
  - complete system, orthogonal idempotents, 313
  - decomposition version 2, 314
  - idempotent, degree 2, 312
  - spine factor algebra, 313
  - theory, 311–312
  - symmetric cones/self-scaled cones, 297 techniques, 298–299, 334 theory, 297, 298
- Fourier–Laplace transform, 423, 424
- Free  $*$ -algebra, 378, 396, 397, 609, 617, 624
- Free polynomial, 292, 378
- Free variables, handling, 477–478
- Full greedy solution, 926–927
- Full matrix algebra, 228, 231, 232

**G**

Gangster constraint, 798, 808  
 Gegenbauer polynomials, 240, 254–256  
 $G$ -endomorphism, 224  
 Generalized Lagrangian dual, 500  
 Generalized Lagrangian relaxation, 500  
 Generalized Problem of Moments (GPM), 12  
 $G$ -homomorphism, 224  
 $G$ -invariant, 222–226, 237–238  
 $G$ -irreducible, 225  
 $G$ -isomorphism, 224, 237  
 Globally rigid, 890  
 Gleichstellensatz, linear, 384  
 $G$ -module, 222  
 Goemans–Williamson hyperplane, 824, 825, 831, 835, 836  
 Gram matrix, 402, 590, 592–594, 883–886, 909, 926, 927, 935  
 Graph automorphism, 136  
 Graph equipartition problem, 802–813  
 Graph of the Euclidean distance matrix, 887, 888–889, 891, 894  
 Graph partition problem, 805, 807, 814  
 Graph realization, 889–890  
 Greedy methods  
     algorithm, 925–926  
     approximate, 927  
     computational complexity, 927  
     full greedy solution, 926–927  
     group action, 222  
     sort and threshold, 926  
 Group, 135–136, 223, 241, 292  
 Group action, 219, 221–228, 235, 239, 261  
 Group orbit, 244  
 Group representation, 229, 233, 242  
 $G$ -spaces, 222, 224, 236, 239  
 $G$ -subspace, 224, 236  
 $G$ -symmetric, 239  
 GUS of  $L$ , 363, 365, 368

**H**

Haar measure, 236  
 Hadamard product, 204  
 Hamiltonian  
     characterization, 602  
     description, 603  
     operator, 625  
 Hamiltonian circuit, 803  
 Hamming distance  
     block codes, 242  
     commutative case, 240  
 Hamming space  
     block codes, 242

generalizations, 248  
 symmetry reduction, 247  
 two-point homogeneous spaces, 240

Hankel matrix, 39, 401  
 Held–Karp bound, 803–804  
 Hexagonal inequalities, 860, 867  
 Hierarchies  
     comparison, 146  
     convex relaxations, 140  
     and  $k$ -tuples  
         moment matrices, 245–247  
         symmetry reduction, 247  
     Lasserre, 145–146  
     Lovász–Schrijver, 141–143  
     LP and SDP relaxations, maximum independent set, 141  
     parametric optimization, 278–279  
     SDP relaxations, 610–611  
     Sherali–Adams, 143–145  
 Hierarchy of SDP problems, 503  
 Hilbert Nullstellensatz, 33  
 Hironaka decomposition, 261, 263  
 HKM direction, 691, 735–737  
 Hollow subspace, 884  
 Homogeneous cone, 322, 365  
 Homogeneous self-dual (HSD) model  
     accuracy, HSDsqlp.m vs. sqlp.m, 746  
     description, 744–745  
     HKM-like search direction, 745  
     Schur complement equation, 745  
     solution, 745–746  
 Homogeneous space, 239–241  
 $\ast$ -Homomorphisms, 231  
 HSD model. *See* Homogeneous self-dual model  
 Hypermetric inequalities, 480

**I**

Ideal, 32–33, 38–49, 56–58, 115–121  
 Idempotent, 175, 229, 312, 313, 315, 316, 319, 321, 327, 328, 343, 365  
 ILP. *See* Integer linear programming  
 Incremental Rank Algorithm (IRA) scheme  
     algorithm, 842  
     based Heuristic, 836  
     description, 832  
     “good vertex”, 835–836  
     implementation, 835  
     iteration, 836  
     principle, 832  
 SDP problem, 831  
 Independence number, 220, 242–243, 246, 251, 252

- Independent set, 40, 48, 145, 154–156, 251, 807
- Independent set constraints, 807
- Indicator function, 476, 490, 495
- Inequalities
- linear matrix, 177, 381
  - odd cycle, 125
  - Pataki's, 71–72
  - polynomial, 53–55
- (Infeasible) central trajectory (or path), 347
- Integer linear programming (ILP)
- vs. CP, 637
  - formulation, MAX-SAT problem, 652
  - model, 650
- Integrality gaps
- approximation hardness, 165–166
  - CSPs, 162–164
  - lower bounds, 159
  - reductions use, 165
  - vertex cover
    - description, 159
    - local distributions, solutions, 160
    - LS hierarchy, 160–161
    - semidefinite programs, 161–162
    - Sherali–Adams hierarchy, 161
- Integral transforms, 423–425
- Interior, 264, 301–303, 325–334, 344–354, 437–453, 471–495
- Interior point map, 346–347
- Interior-point methods (IPMs)
- algorithms and convergence properties
    - homogeneous, 350–352
    - infeasible interior point, 348–350
    - local properties, 353–354
  - cutting-plane algorithms, 478–484
  - cutting plane implementation, 490–491
  - max-cut and facility layout, 491–493
  - perturbations, warm starting, 493–494
  - primal and dual symmetric cone, 345–346
  - self-concordant barrier, 344
  - self-regular, 437
  - semidefinite optimization problem, 345
  - warm-starting strategy, 484–489
- Invariant ring, 261
- Invariant semidefinite programs
- applications
    - combinatorial optimization, 264
    - interior point algorithms, 264
    - low distortion geometric embedding problems, 264
    - polynomial optimization, 264
    - software, 265
- block codes, 242, 248
- Hamming space and distance, 242
  - hierarchies and  $k$ -tuples, 245–247
  - Lovász and Delsarte's linear programming bound, 242–244
  - triples, 244–245
- complex, 220–221
- crossing numbers
- description, 248–249
  - Zarankiewicz's problem, 249–250
- description, 219
- group action
- block diagonalization, 223–227
  - $G$ , group, 221–222
  - $G$ -invariant, 222
  - permutation, 222
  - regular \*-representation, 227–228
  - restriction, invariant subspace, 223
- matrix \*-algebras
- block diagonalization, 231–233
  - commutative, 229–230
  - \*-homomorphisms, 231
  - positive semidefinite elements, 230
  - regular \*-representation, 234
- positive definite functions, compact groups
- commutative case, 239–241
  - compact group  $G$ , 234–235
  - $G$ -invariant, 237–238
  - noncommutative case, 241–242
  - unitary representations, 236
- spherical codes
- applications, 259–260
  - computation, 251–252
  - definition, 250
  - generalization, Lovász, 253–254
  - Lovász, 252–253
  - minimal angular distance, 251
  - orthogonal group, positive definite functions, 254–256
  - reduction using symmetries, 257–259
  - stabilizer, positive definite functions, 256–257
- sums of squares
- description, 260–261
  - invariant theory, 261
  - symmetries, 261–263
- Invariant theory, 261
- Inverse elements, 306–309
- IRA scheme. *See* Incremental Rank Algorithm scheme
- Irreducible representations, 608
- \*-Isomorphism
- block diagonalization, 225–227
  - \*-homomorphism, 231

- \*-Isomorphism (*cont.*)  
     matrix \*-algebras, 228  
     regular \*-representation, 227–228
- Isotypic component, 224, 233, 238
- J**  
   Jacobian matrix, 62, 64, 456  
   Jacobi polynomials, 240, 241  
   Johnson space, 240, 241  
   “Joint + Marginal” and optimization algorithm  
     continuous problem, 288  
     polynomial program, 286  
     0/1 programs, 291–294  
     semidefinite relaxations, 289–291  
   parametric optimization  
     consequences, 280–281  
     global minimizer mapping, 281–282  
     infinite-dimensional linear, 274–276  
   parametric polynomial problem, 271  
   robust polynomial optimization (*see* Robust optimization, “Joint + Marginal”)  
   Jordan algebra, 297–335, 342–345, 359–360, 362, 368  
   Jordan frame, 244, 248, 313, 314, 316–319, 322–324, 326–327, 343, 365, 368
- K**  
   Karush–Kuhn–Tucker (KKT) conditions, 66, 691  
   *k*-equipartition problem (*k*-GP)  
     applications, 805  
     block form in  $Y$ , 807–808  
     definition, 804–805  
     eigenvalue based bounds, 806  
     independent set constraints, 807  
     Laplacian matrix, 805  
     nonnegativity constraints, 809  
     SDP problem *k*-GPzw constraints, 806–813  
     sparsity pattern, 808  
     triangle constraints, 807  
   Kernel function  
     non-self-regular, 451–453  
     SR-proximity measure, 445  
   *k*-GP. *See k*-equipartition problem  
   Kissing number problem, 251  
   KKT conditions, 69, 461, 462, 465, 765, 919  
   K-level polytope, 132  
   Krawtchouk polynomials, 240, 241, 243  
   Kronecker product, 377–378, 380  
   Kronecker sum, 304, 318
- L**  
   Lagrangian, 96, 97, 99, 461, 464, 465, 467, 469, 574, 756, 758, 763, 764, 766, 770, 772, 774, 782, 783, 892  
   Lagrangian dual, 482  
   Lagrangian duality, 574, 583  
   Lagrangian function, 301, 461, 541  
   Lagrangian multipliers, 63  
   Laplacian eigenvalue, 803, 815–816  
   Laplacian matrix, 805  
   Large-scale SDP, 632. *See also* SDPA family developments  
   Large-update SR-IPM, 451  
   Lasserre’s SDP relaxation, 500, 503–506  
   LCP( $L, q$ ), 362, 363, 366, 367–369  
   Left multiplication operator, 303  
   Lie algebra, 334  
   Linear arrangement, 850, 851, 854, 857, 858  
   Linear inequalities, handling, 475–477  
   Linearized cone, 457, 460, 463, 464  
   Linearized quadratic linear ordering problem (LQLO), 855, 856  
   Linear matrix inequality (LMI)  
     domination and complete positivity, 381–388  
     and nc LMI, 377–378  
     representation, convex sets and hulls (*see* LMI representation, convex sets and hulls)  
       representations, 382, 389  
       ubiquitous, 381  
   Linear ordering, 487, 492, 826  
   Linear pencil, 81, 94, 380, 381, 383–385, 388, 389, 395, 401  
   LMI. *See* Linear matrix inequality  
   LMI representation, convex sets and hulls  
     algebraic interiors, 82  
     determinantal representations, 85–86  
     linear engineering systems, 78  
     line test, 77  
     monic, 77  
     properties, 81–82  
     real zero condition, 82  
     result, 84–85  
     rigid convexity, 83–84  
     sets and polynomials, 81  
   Localizing moment matrix, 57, 94  
   Logarithmic barrier function, 441, 442, 542, 547  
   Lovász theta number, 588–589  
   Löwner operator, 355, 358–361, 366  
   Löwner partial order, 2  
   Low Rank SDP formulation (LRSDP), 831, 832

- Low-rank structure, 621–722, 717, 732, 733, 738–739, 841
- LQLO. *See* Linearized quadratic linear ordering problem
- LRSDP. *See* Low Rank SDP formulation
- Lyapunov transformation, 342
- M**
- Marginal, 275. *See also* “Joint + Marginal” and optimization
- MATLAB
- NCSOStools, 402
  - SDPA-M, 697
  - toolbox, 607, 619
- Matrix  $*$ -algebras
- block diagonalization, 231–233
  - commutative, 229–230
  - definitions and examples, 228–229
  - $*$ -homomorphisms, 231
  - positive semidefinite elements, 230
  - regular  $*$ -representation, 234
- Matrix completion
- bound constraints, RBR, 543–545
  - SDP, 553–554
- Matrix lifting, 806
- Matrix positive, 394, 395, 400, 402
- Max-cut
- approximation algorithm, 148–149
  - bipartitions, 823
  - characteristic vector, 134
  - computational results, 824
  - computational status, 841–844
  - extensions
    - Max- $k$ -cut and coloring, 824–826
    - ordering problem, 826–827
  - good solutions, 834–836
  - hyperplane rounding, 148–149
- LP relaxations, 837–841
- primal and dual problem, 822
- problems, 479, 491, 821–847, 850, 852–853, 856, 864
- SDP
- interior-point methods, 828
  - non-linear programming methods, 830–834
  - spectral bundle method, 829–830
- SDP relaxations, 839–841
- semidefiniteness constraint, 823–824
- theta body, 134, 135
- unconstrained problem, 822
- weight edge function, 821–822
- Max-cut LP relaxation
- cut polytope, 837
- cycle inequalities
- cutting plane algorithm, 838
  - separation problem, 838
  - sparse graphs, 838
- denser instances, 838
- facet defining inequalities, family, 838
- vs. SDP relaxation
- BiqMac algorithm, 840
  - bound computation, 839
  - branch and bound algorithm, 839–840
  - technical details, 840–841
- semimetric polytope, 838
- triangle inequalities, 838
- Maximum-cut problem. *See* Max-cut
- Max-cut SDP relaxation
- dense instances, 839
  - denser graphs, 839
  - elliptope, 839
  - vs. LP relaxations (*see* Max-cut LP relaxations)
- Moore–Penrose pseudo-inverse, 541
- PURE-RBR-M, 542–543
- RBR subproblem, 541
- spectrahedron, 839
- Maximal cliques, 509–511, 514, 524
- Maximum bisection problems
- feasible solution, 181–182
  - nonnegative symmetric matrix, 180
  - relaxation, 181
- Maximum entropy, 281
- Maximum  $p$ -section problems, 182
- Maximum stable set problem, 196–197
- Maximum weighted stable set problem
- description and model formulations, 650
  - hybrid approach, 649
- MCNC benchmark problems, 873
- Merit/smoothing function methods
- definition
    - C-function, SCCP, 355–356
    - error bound, SCCP, 355
    - semismoothness, 356
  - SCCP
    - C-functions, 360–361
    - convexity, 361
    - differentiability, 361
    - semismoothness, 361
- SDCP
- Chen and Mangasarian, 358–359
  - description, 357
  - Fischer–Burmeister, 357
  - natural residual function, 357
- SOCCP
- description, 359
  - structure, 360

- Method of augmented Lagrangians, 549–552, 555, 558, 569, 579, 584, 757, 798, 799, 832, 841
- Metric polytope, 479–480
- Min-cut problem, 814
- Minimal idempotent, 229, 230, 232
- Minimal vertex separator, 888–889
- Minimum distance, 242
- Minimum  $k$ -cycle covering problem, 180
- Minimum polynomial, 307, 308, 312–315
- Modified Newton Method, 760–761
- Moment approach, 27, 53, 57, 271
- Moment matrix, 14–15, 30, 36, 39, 41, 93, 155, 276–277, 605, 606, 610, 619, 625
- Moment problem, 41, 400, 409, 411, 432
- Moment property, 410, 411
- Monotone property, 340
- Monteiro-Zhang Family, 332, 349
- MUB. *See* Mutually unbiased bases
- Multiple precision library, 697
- Multiplication matrix, 35, 175, 223, 224, 302–305, 309, 696, 828
- Multiplication parameter, 227, 234, 250
- Multi-thread computation, 701
- Mutually unbiased bases (MUB), 627–631
- N**
- Natural residual function, 357, 358, 360, 361
- NCP, 344, 355, 357, 358, 362
- Neighborhood (of the central path), 488–489
- Nesterov–Todd (NT) direction, 333, 351, 441, 453
- Nesterov–Todd method, 298, 333, 351, 353, 441, 453
- News data
- explain variance, percentage, 930–931
  - principal components, 928
  - thresholded PCA, 930, 932
  - words, sparse PC, 929–930
- Newton system, 474, 477, 486
- NLCP( $\psi, q$ ), 362, 363, 365, 367–369
- NLP methods. *See* Non-linear programming methods
- Non-commutative polynomial optimization
- asymptotic convergence, 614–616
  - commutative case, 622
  - convergence properties, problems, 626–627
  - duals and Positivstellensatz, 616–617
  - equality constraints, 617–618
  - finite vs. infinite dimension, 622–624
  - hierarchy, SDP relaxations (*see* Hierarchies)
  - illustration method, 618–621
  - interacting spin systems, 603–608
  - mutually unbiased bases, 627–631
  - NC polynomial problem, 609
  - optimality detection, 612–613
  - quantum chemistry, 602–603
  - quantum correlation, 601–602
  - unbounded operators, 624–626
- Non-commutative polynomials
- indeterminates, 379–380
  - involution, 379
  - LMI<sub>s</sub>, 380–381
  - matrix-valued polynomials, 380
  - polynomial  $p$ , 379
- Non-commutative real algebraic geometry, 390
- Non-commutative set, 382, 383, 385, 389, 392, 395
- Nonconvex program, 210, 216
- Nondegeneracy condition
- interior point methods, 328–334
  - standard form SCP, 325
  - strict complementarity, 326–328
  - strong duality theorem, 325–326
  - symmetric conic programming, 325
- Nonlinear matrix inequality, 755–789
- Non-linear programming (NLP) methods
- augmented Lagrangian method, 832–833
  - constraint qualifications, 458–460
  - critical directions, cone, 462
  - efficiency, 844
  - IRA scheme, 832
  - Lagrangian, 461
  - linearized cone, 457–458
  - and NLSDPs, 455–456
- Non-linear SDP (NLSDP)
- constraint qualification, 460–461
  - feasible set, 457
  - and NLPs, 455–456
- Nonlinear semidefinite programming problem, 344, 455
- Nonnegative orthant, 487
- Nonnegativity constraints, 806, 807, 809, 816
- Normal form, 26, 30, 34–37, 47, 162
- Normal map, 362, 363
- $N^+(K)$ -relaxation, 798
- NT. *See* Nesterov–Todd
- Nuclear norm, 895
- Null space, 206, 554, 757, 797, 885–886
- Nullstellensatz, 399
- O**
- Objective function, 486, 500, 503, 523
- Octonion, 240, 320, 323
- Odd-cycle inequalities, 480

- Off-central path, 353, 354  
 Online Solver. *See* SDPA family  
     developments  
 Operator commute, 306, 310, 343, 366  
 Optimal value function, 272, 273, 276, 283,  
     284  
 Optimization codes, 575  
 Orbit, 192, 800  
 Ordering problem, 826–827  
 Orthogonality relations, 631  
 Orthogonal polynomials, 239–240
- P**
- Parallel computation, 694, 701–703, 706–708  
 Parametric optimization, 271–283, 287, 288,  
     289  
 Pataki’s inequalities, 71–73  
 Path following interior point method, 907  
 Path-following method, 489  
 PCA. *See* Principal component analysis  
 PDIPM. *See* Primal-dual interior-point method  
 Peirce decomposition, 315–320, 323, 327, 334,  
     343–344, 365, 366  
 Peirce space, 315–318  
 $\ell_0$  Penalization  
     convex maximization problem, 920  
     lower bounds, 922  
     positive square root, 921  
 $\ell_1$  Penalization  
     eigenvalue problem, 919  
     lifting procedure, 918  
 Penalty function, 550, 757, 759, 763, 833, 856  
 Penalty method, 485  
 PENBMI  
     simultaneous stabilization BMIs, 779–780  
     static output feedback problem, 777–779  
     user interface, 776–777  
 PENNON  
     approximation, nonnegative splines,  
         787–789  
     correlation matrix, constrained condition  
         number, 785–787  
     examples, nonlinear semidefinite programs,  
         784–785  
     user interface, 783–784  
 PENSDP, 771–773  
 Pentagonal inequalities, 860, 866  
 Permutation matrix, 508, 509, 795, 799, 826  
 Persistence, 280–281  
 Peter-Weyl theorem, 236  
 Piecewise polynomial, 279, 283, 286,  
     407–408  
 Polarization, 305, 309, 310, 315
- Polynomial equations  
     emptiness, exact certificates, 55–56  
     existing methods, 26  
     moment method  
         coefficient vectors, 29  
         Hankel structure, 30  
         linear system, 29  
         nonreal solutions, 29  
         positive semidefiniteness, 30  
         real radical ideal, 30  
         symbolic/numeric algorithms, 28  
     optimization and polynomial inequalities,  
         53–54  
     positive dimensional ideals and quotient  
         ideals, 56–58  
 real numbers  
     bisection search, 27  
     Khovanskii-Rolle continuation, 28  
     moment methods, 28  
     real roots, 27  
     subdivision methods, 27–28  
 real root finding  
     vs. complex root finding, 51–52  
     moment matrix algorithm, 49–51  
     positive linear forms and real radical  
         ideals, 38–43  
 Polynomial ideals, theta bodies  
     combinatorial moment matrix, 116–117  
     real radical ideals, 118–121  
     types, 115–116  
 Polynomial optimization. *See also* Non-  
     commutative polynomial  
     optimization  
 applications  
     Motzkin’s polynomial, 593–595  
     orthogonality, constraints, 591–592  
     random full-rank polynomial SOS  
         problems, 592–593  
     random low-rank polynomial SOS  
         problems, 593  
     regularization methods, 589  
     regularization vs. projection, 595  
     SOS, SDP and software, 589–590  
     tolerance parameters, 592  
     unconstrained polynomial  
         minimization, 596–597  
 Cholesky factorization and chordal graph,  
     508–510  
 dense SDP relaxation, 523–524  
 equality and constraints, 515–517  
 higher relaxation order, 517–518  
 Lasserre’s dense SDP relaxation,  
     503–506  
 moment approach, 53–55

- Polynomial optimization (*cont.*)
- moments and positive polynomials
    - localizing matrix, 15
    - moments matrix, 14–15
    - positivity certificates, 15–17
  - moment-sos approach, 12
  - numerical results, SFSDP, 526–529
  - optimal solutions, computation, 517
  - primal-dual interior point, 506–508
  - quadratic optimization formulation, 521–523
  - scaling, 518
  - semidefinite programming, 569
  - semidefinite relaxations
    - large size problems, 21
    - nonlinear 0/1 programs, 20
    - software, 21
  - size reduction, 518
  - sparse SDP relaxation, 511–514, 524–526
  - structured sparsity formulation, 510–511
- Polynomials characteristics, 307–308
- Polynomial positivity and optimization
- continuous piecewise polynomials, 425–426
  - elementary semi-algebraic functions, 414–417
  - global optimization problem, 426–427
  - integral transforms, 422–425
  - moment-sos approach, 407
  - non-polynomial functions, 408
  - non semi-algebraic functions, 421–422
  - original optimization problem, 408
  - quadratic modules and polars, 409–414
  - semi-algebraic functions, classes, 417–420
  - sparsity, 428
  - stability, 429–432
  - theoretical principles and results, 408
- Polynomial sum-of-squares (SOS), 14, 25, 29, 30, 42, 54, 114, 119, 394, 424, 500, 590
- Polynomial time algorithm, 139, 166, 471
- Positive definite functions
- commutative case
    - coding theory applications, 241
    - Grassmann spaces, 240–241
    - $G$ -space  $C(M)$ , 239
    - Hamming distance, 240
    - $q$ -Johnson spaces, 240
    - two-point homogeneous spaces, 239–240
  - compact group  $G$ , 234–235
  - description, 235
  - $G$ -invariant, 237–238
  - noncommutative case, 241–242
- orthogonal group, 254–256
- stabilizer, 256–257
- unitary representations, 236
- Positive definite matrix, 71, 120, 354, 441, 546, 554, 737, 740, 742
- Positive PM, property of  $L$ , 363
- Positive polynomial, 14–18, 395, 624
- Positive semidefinite, 230
- Positive semidefinite measure, 253
- Positivity certificates
- continuous piecewise polynomials, 425–426
  - elementary semi-algebraic functions, 414–417
  - general class, semi-algebraic functions, 417–420
  - integral transforms, 422–425
  - non semi-algebraic functions, 421–422
  - semi-algebraic functions, classes, 417–420
- Positivstellensätze
- linear, 384
  - non-commutative polynomial, 396–397
  - and spectrahedron, 388
  - symmetric polynomials, 397
  - theory, 383
  - tracial, 400–401
- Potential reduction interior point method, 329, 678
- Power associative algebra, 315
- P-property of  $L$ , 362–366
- P-property of  $\Psi$ , 73
- P-property of the LCP over  $\mathbb{R}_+^n$ , 362
- Preconditioner, diagonal, 762
- Preconditioner, L-BFGS, 762–763
- Predictor corrector, 475
- Preprocessing, SQLP
- complex data, 728–729
  - diagonal blocks, detection, 728
  - nearly dependent constraints, 728
  - rotated cones, 729–730
  - unrestricted blocks, detection, 728
- Primal degenerate, 397, 896
- Primal-dual interior-point method (PDIPM)
- description, 688
  - framework, 691
  - KKT conditions, 691
  - semidefinite optimization, 447–451, 747
  - theoretical convergence, 693
- Primal-dual method
- augmented Lagrangian, 583–584
- Principal component analysis (PCA)
- algorithms
    - first-order methods, 923–925
    - greedy methods, 925–927

- applications  
 news data, 928–931  
 senate voting data, 931–933  
 stock market data, 933–935
- Lanczos method, 916
- semidefinite relaxations  
 description, 917–918  
 $\ell_0$  penalization, 920–922  
 $\ell_1$  penalization, 918–919
- simple thresholding methods, 916–917
- statistical fidelity, 915
- Principal minor, 8, 344, 362, 363
- Procrustes problem, 900
- Projection methods, conic optimization, 565
- Projective space, 63, 66, 67, 240
- Projective variety, 63, 66, 67
- Proximal algorithm  
 augmented Lagrangian, 583–584  
 convex optimization, 581  
 primal, 582–583
- Proximity measure, 444–447
- $\Psi$ , 73
- $\Psi_I \subset \Psi$ , 73
- Q**
- QAP. *See* Quadratic assignment problem
- $q$ -Johnson space, 240, 241
- Q method, 328, 334
- Quadratically constrained quadratic programming (QCQP), 61, 70
- Quadratic assignment problem (QAP)  
 automorphism groups, 798  
 convex relaxations, 797  
 Koopmans–Beckmann forms, 796  
 $N^+(K)$ -relaxation, 798  
 and quadratic programming bound, 797  
 semidefinite programming relaxations, 189, 797  
 transitive automorphism group, 799–803  
 triangle inequalities, 799
- Quadratic linear ordering (QLO), 855–858
- Quadratic modules, 409–414
- Quadratic optimization formulation of SNL, 521–523, 853–854
- Quadratic program  
 equivalent forms, 479  
 MAX-2-SAT, 653  
 multicast partition problem, 661  
 primal-dual method, 483  
 warm-start perturbation and SDP relaxation, 487
- Quadratic representation, 309–310
- Quadratic unconstrained binary optimization (QUBO)  
 definition, 853–854  
 max-cut problems, 853
- Quantum chemistry  
 electronic structure and properties, 602–603  
 problems, 608
- Quantum information, 628
- Quantum physics, 298, 383
- Quasi-Newton, 575–578
- Quaternion, 240, 300, 302, 304, 305, 308, 319, 323
- Quaternion algebra, 240, 304, 308, 319, 323
- QUBO. *See* Quadratic unconstrained binary optimization
- R**
- Radical ideal, 28, 33, 35, 38–49, 56, 114, 118–121
- Random matrices, 935–936
- Rank minimization, 893–895
- Rank-two updates, 536, 554–556, 561
- Real algebraic geometry  
 projection theory, 418  
 spectacular applications, 432
- Real algebraic variety, 118
- Real Nullstellensatz, 118
- Real radical ideals, 25–60, 118–121
- Real root finding  
 vs. complex root finding, 51–52  
 moment matrix algorithm, 49–51  
 positive linear forms and real radical ideals, 38–49
- Recession cone, 203
- Reformation equation, 355
- Reformulation, 486
- Regular Borel measure, 235, 253
- Regularization algorithm, 584–589
- Regular \*-representation  
 matrix \*-algebras, 227–228  
 reduction, matrix sizes, 227–228
- Relaxation  
 linear (*see* Linear relaxation)  
 SDP (*see* SDP relaxation)  
 semidefinite (*see* Semidefinite relaxations)
- Relaxation order, 500, 503, 517–518
- Relaxation transformation, 365, 366
- Representation theorem  
 Positivstellensatz, 617  
 SOS, 605
- Rewriting family, 37, 38
- Right multiplication operator, 303
- Robustness interpretation, 919

Robust optimization, “Joint + Marginal”, 283–286  
 Row-by-row (RBR) method  
     application, max-cut SDP, 552  
     augmented Lagrangian method, 549–552  
     matrix completion SDP application, 553–554  
     prototype, 538–540  
     RBR-MC, 558–560  
 $r$ -realization, 889  
 Running intersection property, 408, 428

**S**

Sandwich theorem, 252, 588  
 Scaling of polynomial optimization problems, 61, 264, 499–541, 688  
 SCCP. *See* Symmetric cone complementarity problem  
 SCCP properties  
     Cartesian P properties definition, 367–368  
     complementary cone, L, 369  
     P-property  
         definition, 362  
         implications, L, 363–365  
         implications, 365  
     LCP, 362–363  
     NLCP, 366–367  
         transformation  $R\varphi$ , 366  
     sufficient properties, L, 367  
     uniform nonsingularity, 368–369  
 Schur complement. *See* Schur complement matrix  
 Schur complement matrix computation  
     quadratic and linear blocks, 739–740  
     semidefinite blocks, 692, 738–739  
 equation  
     analytical expression, 740  
     Cholesky factorization, 740–741  
     SQMR method, 741  
     specialized routine, 743–744  
 Schur’s lemma, 225, 233  
 SCM. *See* Schur complement matrix  
 SCP. *See* Symmetric conic programming  
 SDP. *See* Semidefinite programming  
     SDPA family developments, 688  
     SDPA, 697  
     SDPA-C, 698  
     SDPA-GMP/DD/QD, 697–698  
     SDPA-M, 697  
     SDPARA, 700  
     SDPA Online Solver, 704  
     SDPARA-C, 703

SDPA format, 675, 679  
 SDPA input file, 675–681, 687–723  
 SDP relaxation. *See also* Combinatorial optimization problems; Non-commutative polynomial optimization  
     computational performance, 873–874  
 CP search, 647  
 distribution, solution values, 654–655  
 lower and upper bounds, 654, 655  
 non-overlap constraints, 871  
 transitivity property, 872  
 SDPT3 implementation and usage  
     algorithm, 722–730  
     cell array representation, 718–722  
     computational testing, SDPT3-4.0, 750–751  
     feasibility detection, 746–749  
     HSD model, 744–746  
     modelling languages, 730–731  
 Second order conditions constraint qualifications, 458–460  
 Second-order cone, 202, 203, 212, 213, 495, 536, 539, 541, 551, 554, 555, 567, 679, 715, 716, 729, 730, 751, 851, 868, 869, 871, 881  
 Second order cone complementarity problem (SOCCP), 344, 356, 359–360  
 Second order conic programming (SOCP)  
     codes  
         CPLEX, 684  
         LOQO, 680–681  
         MOSEK, 680  
         SDPT3, 678–680  
         SeDuMi, 676  
         SMCP, 683  
 Second order conic optimization (SOCO), 453  
 Second-order cut, 481–482  
 Second order necessary and sufficient conditions, 465–469  
 SeDuMi format, 675  
 Self-concordant barrier, 474  
 Self-Regular (SR)  
     functions, 442–444  
     Primal-Dual IPMs, 447–451  
     proximity measures, 444–447  
 Self-Regular Interior-Point Methods (SR-IPMs), 437  
 Semi-algebraic functions  
     classes, 417–420  
     elementary, 414–417  
     non semi-algebraic functions, 421–422  
     optimization, 426–428  
 Semi-algebraic sets

- convex cone, 577–578  
 non-commutative convex  
   convex semi-algebraic sets, 390  
   LMI representation, 389  
   motivation, 391–392  
   non-commutative basic open  
     semi-algebraic sets, 390  
 polynomial optimization, 569
- Semidefinite and polynomial optimization  
 algebraic degree derivation  
   algebra geometric methods, 66–68  
   complex projective setup, 63–65  
   degree formulas, 68–75  
   genericity assumptions and KKT, 65–66
- optimization problem  
 affine variety, 62  
 Jacobian matrix, 62  
 Lagrangian multipliers, 63  
 polynomial functions, 62  
 semidefinite programming, 63
- Semidefinite complementarity problem  
 (SDCP), 675–676
- Semidefinite cone, 2, 201, 202, 204, 205, 212, 213, 357, 368, 465, 558, 561, 579, 715, 902
- Semidefinite cut/cutting, 482, 494, 683
- Semidefinite Optimization (SDO). *See*  
 Self-Regular Interior-Point  
 Methods (SR-IPMs); Semidefinite  
 programming (SDP)
- Semidefinite program, complex, 220–221
- Semidefinite programming (SDP)  
 algebraic degree, 71–72  
 alternating direction method, 573  
 application  
   matrix completion, 553–554  
   max-cut SDP, 552  
 approach of moments, 25–60  
 augmented Lagrangian method, 549–552  
 BCD method, 534–536  
 combinatorial problems, 6  
 computational complexity  
   ellipsoid algorithm, 11  
   exponential bitlength, 11  
 convergence results, 546–548  
 correlation matrix problem, 578  
 description, 71, 533  
 duality  
   positive duality gap, 10  
   Slater's constraint qualification, 10  
   weak infeasibility, 10
- heuristic accuracy  
 change, satisfied clauses, 656–657  
 distribution, solution values, 654–655
- lower and upper bounds, 654, 655  
 uninformative and informative values, 655–656  
*vs.* Walksat, 657
- Laplacian matrix, 4
- linear matrix inequality (LMI), 2
- Lovász theta number, 588
- matrix completion, 543–545
- max-cut problem, 2–3
- max-cut SDP relaxation, 540–543
- model, 660–662
- notation and organization, 537
- numerical results  
 matrix completion, 558–559  
 max-cut SDP relaxation, 556–558
- Pataki's inequalities, 71–72
- polynomial optimization, applications, 589–590
- primal-dual, 9–10, 690
- rank-two updates, 554–556
- RBR method prototype, 538–540
- relaxation, 5–6
- representation, convex sets and hulls  
 conditions, 90–91  
 convex hulls, semialgebraic sets, 88–90  
 description, 78  
 semialgebraic sets, 86–87
- Schur complement, 538
- sparse inverse covariance estimation, 545–546
- stratification, definition, 71
- Semidefinite programming (SDP) relaxations.  
*See also* Max-cut SDP relaxations  
 association schemes, 176–178  
 benefits, 663  
 coefficient vectors, 19  
 coherent configurations  
   combinatorial optimization problems, 192–194  
 GAP, 183  
 linear matrix inequality, 187–188  
 maximum  $(k,l)$ -cut problem, 194–195  
 maximum stable set problem, 196–197  
 polytope, 184  
 projection, 185–186  
 QAP, 189–192  
 vertex separator problem, 197
- constraint programming  
 Boolean satisfiability problem, 648–649  
 optimization constraint, 647  
 search heuristic, 647–648  
 weighted CSPs, 649
- duality, 892–893

- Semidefinite programming (SDP) relaxations  
 $(cont.)$
- global optimization problem, 18
  - Lagrangian, 892
  - moment and localizing matrices, 20
  - search heuristic, accuracy
    - experimental evaluation, 654–657
    - MAX-2-SAT problem, 651–652
  - semidefinite program, definition, 18
- Semidefinite-quadratic-linear programming (SQLP). *See* SDPT3 implementation and usage
- Semidefinite representation, convex sets and hulls
- constructions and theory, 79
  - LMI representation
    - conditions, 81–84
    - determinantal, 85–86
    - linear engineering systems, 78
    - line test, 77
    - monic, 77
    - sets and polynomials, 81
  - localization, 104–108
  - moment type constructions
    - Krein–Millman theorem, 91
    - linear functional, 92
    - moment matrix, 92–94
    - refined, 94–95
    - SDP representation, 91–92
  - motivation, 79–80
  - notations, 80
  - positive curvature
    - definition, 108–109
    - implicit function theorem, 109–110
    - linear coordinate transformation, 109
    - nonsingular hypersurface, 110
    - second fundamental form, 110
  - SDP representation
    - conditions, 90–91
    - construction, 87
    - convex hulls, semialgebraic sets, 88–90
    - description, 78
    - semialgebraic sets, 86–87
  - semialgebraic sets, 79
  - SOS, 95–98
  - strictly convex sets, 98–104
- Semismooth
- generalized Newton, 577–578
  - Newton-like methods, 575
- Senate voting data, 931–933
- Sensor network localization (SNL) problem
- anchors, 899–901
  - Biswas–Ye semidefinite relaxation, 880, 903–905
- description, 898–899
- NP-hard, 901
- semidefinite relaxation, 901
- sensor positions, 906
- unique localizability, 905–906
- Separating inequality, 481
- Sequential semidefinite programming, 464, 470
- SFSDP, 521, 526–529, 880
- Shifted-barrier method, 485, 487
- Simple  $*$ -algebra, 411
- Single-row facility layout problem (SRFLP)
- betweenness model, 859–861
  - binary quadratic optimization and maximum-cut problem, 853–854
  - distance polytope, 852
  - matrix-based formulations and SDP relaxations, 861–864
  - max-cut problem, 852–853
  - quadratic linear ordering polytope, 855–856
  - quadratic linear ordering problem, 856–858
  - SDP-based heuristic, 864–865
- Singular locus, 62, 73
- Slack variables, 216, 567, 583, 783
- Small update SR-IPM, 441, 445, 447–451, 453
- Smoothing function, 354–361
- SNL. *See* Sensor network localization
- SOCO. *See* Second order conic optimization
- SOCP. *See* Second order conic programming
- Software. *See* Conic optimization software
- SOS. *See* Sum-of-squares
- SOS polynomials
- convex sets, 95
  - strictly
    - concave defining polynomials, 99–101
    - quasi-concave defining polynomials, 103
- Space of  $n \times n$  real symmetric matrices, 14
- Sparse
- $LU$  factorization, 741
  - matrix representation, 719
  - quadratic and linear blocks, 739–740
  - SOCP problems, 741
  - SQLP problem, semidefinite block, 718
- Sparse Cholesky factorizations
- and chordal graph, 508–510
  - MUMPS library, 702, 709
  - SDPA, 675
- Sparse POP format, 519–520
- Sparse Schur Completion Matrix, 688
- Sparse SDP relaxation
- Cholesky factorization and chordal graph, 508–510
  - formulating structured sparsity, 510–511

- POP, 511–514  
 semidefinite programming problems,  
   506–508  
 Sparsity pattern, 798, 808, 812  
 Sparsity pattern matrix, 508, 509  
 Spectral bundle method, 829–830  
 Spectral decomposition, 313, 314, 334, 343,  
   366, 566  
 Spectral decomposition (version 1), 324  
 Spectral decomposition (version 2), 324  
 Spectral function, 323–325, 330  
 SpeeDP algorithm, 844  
 Spherical cap, 251, 259  
 Spherical codes. *See* Invariant semidefinite  
   programs  
 Spherical function, 239–241  
 Spin factor algebra, 308, 310, 313, 314,  
   318–320  
 Spline, 407, 425, 787–789  
 SR. *See* Self-Regular  
 SRFLP. *See* Single-row facility layout  
   problem  
 Stability, differential equations, 429–432  
 Stable set, 122–126, 196–197, 649–651  
 Stable set problem. *See* Maximum weighted  
   stable set problem  
 Standard quadratic programming, 214, 215  
 Static output feedback, 777–779  
 Statistical consistency *vs.* computational  
   complexity  
   greedy algorithms, 936  
   ROC curves, 937, 938  
   thresholding method, 937  
 Stickelberger’s theorem, 26, 35  
 Stock market data  
   explained variance, 934, 935  
   returns project, 934–935  
 Strict complementarity, 476–477  
 Strong duality, 439–441, 473  
 Strongly infeasible, 351, 724  
 Strongly regular graph, 176, 180  
 Structural parameters, 227  
 Structured sparsity, 500, 501, 510–511  
 \*-Subalgebra, 231–233  
 Sufficient conditions, 465–469  
 Sufficient property of  $L$ , 43, 82, 85, 89, 99,  
   162, 174, 237, 310, 356, 362, 694  
 Sum-of-squares (SOS)  
   approach of moments, 29, 30, 50, 55–56  
   convex sets, 95–98  
   decompositions, 616, 617, 620  
   hereditary polynomials, 625  
   invariant semidefinite programs, 260–263  
   orthogonality, constraints, 591  
   random full-rank polynomial, 592–593  
   random low-rank polynomial, 593  
 SDP and Software, 589–590  
 Sum of squares polynomial, 14, 42, 114, 118,  
   394, 500, 590  
 Symmetric cone, 297, 323, 325, 326, 342  
 Symmetric cone complementarity problem  
   (SCCP), 339  
 Symmetric conic programming (SCP),  
   325–329  
 Symmetric function, 324, 325, 396  
 Symmetric set, 324  
 Symmetry  
   reduction, 257–259, 799, 800, 804, 811,  
   813  
   sums of squares, 261–263  
 Symmetry reduction, 257–259
- T**
- Tangential cones  
   constraint qualifications, NLP, 458–460  
   feasible set, 457–458  
 Tensor product, 78, 173, 174, 228–229, 624  
 Tent function, 425, 426  
 Terwilliger algebra, 244  
 Theta bodies computation  
   examples, 121–122  
   infinite variety  
     cardioid, 127  
     polynomial, 126  
      $\text{TH}_2(I)$ , 128, 129  
 maximum stable set problem  
   discrete optimization, 123–124  
   odd-cycle inequalities, 125–126  
   polytope, 122  
   semidefinite program, 124  
 Topological dual, 253  
 Tracial moment sequence, 400, 401  
 Transitive automorphism group, 799–803  
 Traveling salesman and the  $k$ -cycle covering,  
   179–180  
 Traveling salesman problem (TSP), 803–804  
 Triangle inequalities, 479–480, 491–492, 799,  
   838  
 Trigonometric function, 407, 424  
 Trigonometric polynomial, 408, 432  
 Two-point homogeneous space, 239–241
- U**
- Unconstrained binary quadratic program, 834  
 Uniform nonsingularity, 368, 369  
 Unique localizability, 905–906

Unitary representations  
 compact groups, 236  
 definition, 222  
 group action, 221  
 matrix \*-algebras, 224  
 set, invariant matrices, 229  
 Universally rigid, 890  
 Unrestricted variables, 717, 722–723, 728, 744

**V**  
 Variety, 31–33, 62, 63, 113, 126–128  
 Vector lifting, 806  
 Vertex separator problem, 197  
 Very large scale integration (VLSI)  
   floorplanning  
   mixed-integer conic optimization  
     formulation, 868, 871  
   SDP relaxation, 871–874  
 Volumetric center, 481

**W**  
 Warm start  
   cutting-plane schemes, 487–488  
   feasibility restoration, perturbations,  
     488–489  
   interiority, shifted barriers,  
     485–487  
 Weak duality, 439, 473

**Y**  
 YALMIP, 619, 705, 706, 730–731,  
 779  
**Z**  
 Zarankiewicz’s problem, 249–250  
 Zariski closure, 64  
 Zero algebra, 228  
 Zonal spherical function, 239