LINEAR PROGRAMMING APPROACHES TO SEMIDEFINITE PROGRAMMING PROBLEMS

By

Kartik Krishnan Sivaramakrishnan

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Mathematics

Approved by the
Examining Committee:

John E. Mitchell, Thesis Adviser

Kristin Bennett, Member

Mark Goldberg, Member

Thomas Yu, Member

Rensselaer Polytechnic Institute
Troy, New York

July 2002
(For Graduation August 2002)

LINEAR PROGRAMMING APPROACHES TO SEMIDEFINITE PROGRAMMING PROBLEMS

By

Kartik Krishnan Sivaramakrishnan

An Abstract of a Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Mathematics

The original of the complete thesis is on file
in the Rensselaer Polytechnic Institute Library

Examining Committee:

John E. Mitchell, Thesis Adviser
Kristin Bennett, Member
Mark Goldberg, Member
Thomas Yu, Member

Rensselaer Polytechnic Institute
Troy, New York

July 2002
(For Graduation August 2002)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

The title of your thesis doesn't matter. The subject doesn't matter. The research doesn't matter. All that matters is who your advisor is.
– The Rabbit's Thesis

First and foremost, I wish to express my deep sense of gratitude towards my advisor John Mitchell on matters technical and otherwise, for all the things I've learned from him, and the various colloborations we've had over these years at RPI. While in the last four years I've found RPI to be every bit the wonderful place I anticipated it to be and more, I'd still not be exaggerating if I said that my education at RPI might not have been complete, enjoyable and fulfilling, had I not had John as my supervisor. He was always there to answer my questions, and patiently persevered through various versions of this manuscript, considerably improving the quality of exposition of the thesis. Although I am sure he will disagree, John is one of the foremost exponents of interior point branch and cutting planes schemes for linear programming problems; the ideas in chapters 3 and 4 were largely influenced by his various insights on the subject. All in all, it's been a privilege John. Thank you!.

I wish to thank the other members on my thesis committee : Kristin Bennett, who incidentally taught me my first course on nonlinear programming, Mark Goldberg, and Thomas Yu. I also thank Thomas for sparing valuable time and help, during my arduous search for postdoctoral and academic positions. My special thanks to the staff in the Math department, in particular Michele Kronau, Lorraine Pisarczyk, Melissa Reardon, and Dawnmarie Robens, for their good cheer, and administrative help without which I wouldn't have possibly graduated. Michele was amazingly efficient with processing my several reimbursement requests, which was very helpful considering my meager bank balance at times.

I wish to thank my family, especially my father N.K. Sivaramakrishnan, my mother Dr. Vasantha Sivaramakrishnan, my brother Sivaraj, and sister-in-law Ramya for their selfless sacrifice, and their eternal spring of inspiration in all my

endeavors. They *always* had confidence in me, even on occasions when I myself didn't. It is only fitting that I am able to emulate my mother, to become the second doctor in the family. I proudly dedicate this thesis to all of them.

I would like to thank the following people for their generous support and encouragement throughout my Ph.D., and stint at RPI : Kristin Bennett, Mark Goldberg, Isom Herron, Gregor Kovacic, Mukkai Krishnamoorthy, Mike Kupferschmidt, Joyce McLaughlin, John Mitchell, Michael Overton, Don Schwendeman, Tamas Terlaky, Henry Wolkowicz, Thomas Yu, and Yin Zhang.

I thank Christoph Helmberg for making his spectral bundle code available, and for useful discussions in Chapter 2. I also thank Beresford Parlett for his illuminating suggestions in improving the lower bounds for the cutting plane LP approach, in Chapter 3.

Last and definitely not the least, I wish to thank my various friends on and off campus for their support, cooperation, and more importantly making my stay at RPI (miles from home!) a pleasant experience. This count is especially large, and a partial list includes Harsh Oke, Grigorios Pavliotis, Ibrahim Fatkullin, Lucian Basescu, Steve Braun, K.G. Rajesh, Navnit Agarwal, Lilia Krivodonova, Xiaoyun Ji, Rohan Kelkar, Darryl Ahner, Sava Dediu, Ajit Srivastava, Maria Reznikoff, and Rafail Abramov. I am sure I am missing out on a lot of names, and my sincere apologies to all those I've missed. I also thank the other members (optimizers!) of our weekly optimization seminar, which included Luc Basescu, Kristin Bennett, Jinbo Bi, Steve Braun, Sava Dediu, Xiaoyun Ji, Abigail Michaels, Michinari Momma, and Jufeng Peng, for the enlightening Friday morning discussions on optimization over a cup of coffee.

I also gratefully acknowledge the teaching and research assistantships that supported my research at RPI. In particular, this research was supported in part by NSF grant number CCR-9901822.

ABSTRACT

Semidefinite Programming (SDP) has been one of the most exciting and active areas in optimization lately. This tremendous activity was spurred by the discovery of efficient interior point algorithms for solving SDP problems, and important applications of the SDP in developing approximation algorithms for various combinatorial optimization problems. However these applications require effective techniques, to solve SDP's arising as relaxations of these problems. Although interior point methods are a great theoretical tool, and offer polynomial time complexity, they are fairly limited in the size of problems they can handle. The thesis investigates linear programming approaches to solving SDP's. This potentially allows one to solve large problems approximately and quickly, using state of the art linear solvers. The LP approach is also incorporated in a branch and cut approach to solving integer programming problems.

One of the various characterizations of the positive semidefiniteness constraint leads to a semi-infinite LP formulation for the SDP. We formulate the dual SDP as a semi-infinite LP, and discuss the issue of its discretization in detail. Using the notions of nondegeneracy and basic feasible solutions developed in the context of semi-infinite linear programming, we recover a theorem due to Pataki on the rank of extreme matrices in SDP, which in turn implies that not more than $O(\sqrt{k})$ (k is the number of constraints in the SDP) constraints are required in the LP relaxations. To generate these constraints we use the spectral bundle approach due to Helmberg and Rendl. This scheme recasts any SDP with a bounded feasible set as an eigenvalue optimization problem. These are convex but nonsmooth problems that can be tacked by bundle methods for nondifferentiable optimization. The LP approach provides an upper bound, and can also be utilized to generate a primal feasible matrix X for the spectral bundle approach. We demonstrate the efficiency of the approach on two combinatorial optimization problems namely maxcut and minbisection.

The semi-infinite LP formulation of the SDP, together with its polynomial

time separation oracle leads naturally to a cutting plane LP approach for the SDP. We investigate such an approach. However to make the resulting method competitive with interior point methods for the SDP, several refinements are necessary. In particular the cutting plane method requires solving the LP relaxations approximately using an interior point method, since this results in better cutting planes. We experiment with various separation oracles generating deep cuts, and techniques to restart the LP relaxations with strictly feasible starting points. We also relate these cutting planes to the geometry of the SDP cone. We then test the approach on the maxcut SDP.

Finally we incorporate the cutting plane LP approach to the SDP in an SDP approach for the maxcut problem. In particular, we formulate the dual of the well known SDP relaxation for maxcut as a semi-infinite LP, and solve it using an interior point cutting plane scheme. We then add cutting planes, which are facets of the maxcut polytope, to the primal problem in order to tighten the SDP relaxations. The approach resembles a SDP cutting plane scheme for the maxcut problem; in reality it uses an LP subroutine to solving the SDP within an overall LP cutting plane approach for integer programming. This overcomes another shortcoming of an SDP cutting plane scheme, where there is no convincing warm start strategy, that allows one to quickly reoptimize a slightly perturbed version of the original SDP, after the addition of cutting planes. We present computational results on a variety of maxcut problems.

To

my parents

N.K. Sivaramakrishnan and Dr. Vasantha Sivaramakrishnan

CHAPTER 1
The Semidefinite Programming (SDP) problem

1.1 Introduction

Semidefinite Programming (SDP) has been one of the most exciting and active research areas in optimization recently. This tremendous interest was spurred by the discovery of important applications in combinatorial optimization and control theory, the development of efficient interior point algorithms for solving SDP problems, and the depth and elegance of the underlying optimization theory.

Semidefinite Programming (SDP) is concerned with choosing a symmetric matrix to optimize a linear function subject to linear constraints and a further crucial constraint that the matrix be positive semidefinite. It thus arises from the linear programming problem (LP) by replacing the vector of variables with a symmetric matrix and replacing the non-negativity restriction on the vector of variables, with the requirement that this symmetric matrix be positive semidefinite. SDP has other similarities with LP too. It is convex, has a rich duality theory (although not as strong as LP's), and also most interior point methods for LP, can be transformed into interior point methods for SDP. There are some difficulties though and we shall examine these later.

Semidefinite programming has its origins in the theory of linear matrix inequalities (LMI's) developed in the Soviet Union in the '40s,'50s,'60s. In the early 1970's Donath and Hoffman [31], and then Cullum, Donath and Wolfe [26] showed that NP hard graph partitioning algorithms could be solved as eigenvalue optimization problems - as we shall see, these are closely connected with SDP. In 1979 Lovasz [78] formulated an SDP problem, that provided an upper bound on the Shannon capacity of the graph, and thereby found the capacity of the pentagon, solving a long open conjecture. Shor in [109] introduced semidefinite relaxations of combinatorial optimization, and non-convex optimization problems.

However, the key contributions to SDP came from Nesterov and Nemirovskii [90], who introduced the theory of self concordant barrier functions, which provided a

general framework for solving various nonlinear optimization problems including the SDP in polynomial time (also see Renegar [104] for a more accessible introduction to this theory), and Alizadeh [1] who showed that interior point methods pioneered by Karmarkar [62] for LP could be extended to SDP. Finally, Goemans and Williamson [36] showed that the SDP relaxation could provide a provably good approximation to the max cut problem in combinatorial optimization (incidentally this SDP relaxation is based on Shor's [109] relaxation of the maxcut problem).

Excellent references for the SDP include the survey paper by Vandenberghe and Boyd [115] which discusses a number of applications, we describe three of these in section 1.5, the book by Boyd et al [19] which describes SDP with regard to control theory, the book by Nesterov and Nemirovskii [90], the paper by Alizadeh [1], the survey paper by Lewis and Overton [76] on eigenvalue optimization, the SDP handbook edited by Wolkowicz et al [118], Helmberg's habilitation thesis [45], Todd's [110] survey paper on semidefinite optimization, and recent monographs by Renegar [104], Ben Tal and Nemirovskii [14], De Klerk [28], and finally [27] edited by El Ghaoui and Niculescu. We should also mention the excellent websites by Helmberg [49], *Optimization Online* [91], and Wright's *Interior Points Online* on semidefinite and conic programming, that allows one to keep abreast of recent developments in this exciting area of optimization. There are also a plethora of software devoted to the SDP : a list of these can be found in Helmberg [49], and Mittlemann [86]. There is also a repository for SDP problems at Borchers' SDPLIB [17], and DIMACS [102]. A recent implementation challenge [102] for solving large scale SDP's, and to benchmark the SDP solvers to date was organized by DIMACS in November 2000. The research in semidefinite programming has literally exploded in the late '80s and '90s. The recent handbook on semidefinite programming edited by Wolkowicz et al [118] lists 877 references, almost all since 1990.

We present an overview of SDP in this chapter. The chapter is organized as follows : Section 1.2 describes the transition from linear programming to general conic programming; this serves to motivate the SDP an important problem in this class, we describe some linear algebra preliminaries on positive semidefinite matrices in section 1.3, section 1.4 introduces the semidefinite programming problem and

studies its duality theory and geometry, we present three applications of the SDP in section 1.5, interior point methods for the SDP in section 1.6, large scale approaches to the SDP in section 1.7, and finally our contribution : a linear programming formulation for the SDP, and the organization of the chapters in this thesis in section 1.8. Finally, a glossary of all the notation in this chapter can be found in Appendix A.

1.2 The transition from Linear Programming to Conic Programming

We assume that the reader is familiar with linear programming (LP). In this case we discuss the transition from LP to convex optimization problems. Nesterov and Nemirovskii [90] show that any convex programming problem is equivalent to minimizing a linear objective function over a particular cone. Hence it suffices to consider the generic conic programming problem. A good reference for most of the material in this section is Ben Tal and Nemirovskii [14].

We begin with a cone. A set C is a cone such that the *open* half line $\{\alpha x : \alpha > 0\}$ is entirely contained in C, whenever $x \in C$. However we will find it useful to work with a cone that is convex as well. Here is the formal definition.

Definition 1 *A set $C \subseteq \mathcal{R}^n$ is a cone if it is closed under non-negative multiplication, and addition i.e. $x, y \in C \Rightarrow \lambda(x + y) \in C, \forall \lambda \geq 0$.*

Consider the generic LP problem

$$\begin{array}{lrcll} \min & c^T x & & & \\ \text{s.t.} & Ax & = & b & (LPP) \\ & x & \geq & 0 & \end{array}$$

with dual

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & A^T y + s & = & c & (LPD) \\ & s & \geq & 0 & \end{array}$$

Here, $c, x, s \in \mathcal{R}^n$, $b \in \mathcal{R}^k$, and $A \in \mathcal{R}^{k \times n}$, with $k < n$. There are two ways

to proceed from LP to its nonlinear programming (NLP) counterpart. Hereafter we will assume that whenever we mention NLP, we are talking about a convex programming problem. The traditional way is to allow the objective function, and constraints to be nonlinear. In contrast to this, we intend to keep the objective and the constraints linear, but introduce nonlinearity in the inequality sign $\geq$.

A significant part of the nice features of LP programs comes from the fact that the ordering $\geq$ satisfies the following properties of an equivalence relation.

1. **Reflexivity :** $a \geq a$.
2. **Antisymmetry :** $a \geq b$, and $b \geq a \Rightarrow a = b$.
3. **Transitivity :** $a \geq b$, and $b \geq c \Rightarrow a \geq c$.
4. Compatible with linear operations : $a \geq b$, and $c \geq d \Rightarrow a + \lambda c \geq b + \lambda d$, for $\lambda > 0$.

We denote any ordering which satisfies axioms 1-4 as $\succeq$. We want to characterize the set $K = \{a \in \mathcal{R}^m | a \succeq 0\}$. This set K cannot be arbitrary; in fact it is a *pointed convex cone*. To see this, observe that for $a, a' \in K$, $\lambda a + a' \in K$, thereby showing that K is a cone, also K is pointed since $a \in K$, and $-a \in K$ imply that $a = 0$, showing that K does not contain any straight lines passing through the origin. Also, define K^* as

$$K^* \quad = \quad \{y \in \mathcal{R}^n : x^T y \geq 0, \forall x \in K\}$$

Here are three interesting cones together with the partial ordering that generates them.

1. The non-negative orthant ${\mathcal{R}_+}^n = \{x \in \mathcal{R}^n : x_i \geq 0, i = 1, \ldots, n\}$.
2. The second order (Lorentz, ice cream) cone

$$\mathcal{L}^n \quad = \quad \{x \in \mathcal{R}^n : x_n \geq \sqrt{\textstyle\sum_{i=1}^{n-1} {x_i}^2}\}$$

3. The cone of positive semidefinite matrices

$${\mathcal{S}_+}^n \quad = \quad \{A \in \mathcal{S}^n : x^T A x \geq 0, \forall x \in \mathcal{R}^n\}$$

The nonnegative orthant (LP cone) is a special case of the second order cone : Note that $x_i \geq 0$ is a second order cone of dimension 1, and hence the nonnegative orthant can be written as the intersection of n second order cones of dimension 1.

On the other hand the second order cone too is a special case of the SDP cone. To see this notice that

$$x_n \geq \sqrt{\sum_{i=1}^{n-1} {x_i}^2} \quad \Leftrightarrow \quad \begin{pmatrix} x_n I_{n-1} & x(1:n-1) \\ x(1:n-1)^T & x_n \end{pmatrix} \succeq 0$$

This follows from the Schur complement characterization in Theorem 1. The cones mentioned above also possess the following useful properties :

1. The cone is closed, i.e. the limit a of a sequence of vectors a^i belonging to the cone, is also in the cone.

2. The cone is full dimensional with a nonempty interior, i.e. there exists a vector such that a ball of radius ϵ centered around this vector, is also contained in the cone.

3. The cone is self dual, i.e. $K = K^*$. We shall show this for the special case of the SDP cone in section 1.3.1.

4. These cones are also special cases of symmetric cones : see Alizadeh and Schmieta [106, 5]), who relate them to Euclidean Jordan algebras. In fact there are only *five* such classes of symmetric cones (Alizadeh and Schmieta [5]). This characterization as symmetric cones allows an easy extension of interior point methods for LP, to convex optimization problems over other cones in this class (Alizadeh and Schmieta [106]).

Interestingly, K satisfies all the above properties if and only if its dual K^* does so too. Given K a cone in $\mathcal{R}^n$ (say), which is convex, pointed, closed, and with a non-negative interior, we can define a conic programming problem as

$$\begin{array}{lrcll} \min & c^T x & & & \\ \text{s.t.} & Ax & = & b & \qquad (CPP) \\ & x & \succeq_K & 0 & \end{array}$$

with dual

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & A^T y + s & = & c & (CPD) \\ & s & \succeq_{K^*} & 0 & \end{array}$$

The details on the derivation of the dual can be found in Ben Tal and Nemirovskii [14], and Renegar [104]. We shall derive the dual through Lagrangian relaxation for SDP case in the next section. Despite the apparent similarity of (CPP) and (CDD) to (LPP) and (LDD), the duality theory in conic programming is not as *strong* as that of the LP case. In the next chapter we shall discuss duality theory in the SDP in more detail, and point out pathologies that may occur in general conic programming.

1.3 Linear Algebra Preliminaries

Having described the transition from LP to general conic programming, we are ready to discuss the SDP. It is first handy to present some linear algebra preliminaries on positive semidefinite (psd) matrices, and their interesting properties. Two excellent references include the books by Horn and Johnson [58, 59].

1.3.1 Positive semidefinite matrices

In the following section we work with the space of symmetric matrices $\mathcal{S}^n$, which is a subset of $\mathcal{M}^n$; the space of square matrices of size n. A matrix $A \in \mathcal{S}^n$ satisfies $A = A^T$. Also, $\mathcal{S}^n$ a vector space in $\mathcal{R}^{\frac{n(n+1)}{2}}$. To see this, consider writing a matrix as one long vector, with the columns stacked end to end. This should give a vector in $\mathcal{R}^{n^2}$. However, since the matrix is symmetric, we need to consider only the upper triangular portion of this matrix, and thus every matrix in $\mathcal{S}^n$ is isomorphic to a vector in $\mathcal{R}^{\frac{n(n+1)}{2}}$. We shall also use the following notation frequently in the thesis.

$$\begin{array}{rcl} n^{\bar{2}} & = & \binom{n+1}{2} \\ & = & \frac{n(n+1)}{2} \end{array} \tag{1.1}$$

An important characterization of a matrix $A \in \mathcal{S}^n$ is its spectral (eigenvalue) decomposition

$$\begin{aligned} A &= P\Lambda P^T \\ &= \sum_{i=1}^{r} \lambda_i p_i p_i^T \end{aligned} \tag{1.2}$$

Here $P \in \mathcal{R}^{n\times r}$ is an orthonormal matrix $P^TP = I_r$, containing the eigenvectors corresponding to the r nonzero eigenvalues of A. $\lambda \in \mathcal{S}^r$ is a diagonal matrix containing these r eigenvalues. Also, r is the rank of A.

A matrix $A \in \mathcal{S}^n$ is symmetric positive semidefinite if $A = A^T$ and

$$d^TAd \geq 0 \quad \forall d \in \mathcal{R}^n \tag{1.3}$$

We denote this by $A \succeq 0$ and the set of symmetric positive semidefinite matrices by ${\mathcal{S}_+}^n$. A matrix $A \in \mathcal{S}^n$ is symmetric positive definite if

$$d^TAd > 0 \quad \forall d \in \mathcal{R}^n \backslash \{0\} \tag{1.4}$$

We denote this by $A \succ 0$ and the set of symmetric positive definite matrices by ${\mathcal{S}_{++}}^n$. We have

$${\mathcal{S}_{++}}^n \subset {\mathcal{S}_+}^n \subset \mathcal{S}^n \subset \mathcal{M}^n$$

Note that we require every psd matrix to be symmetric in the definition.

We also introduce some other terminology that will be useful later. $\mathrm{diag}(X) \in \mathcal{R}^n$ is a vector whose components are the diagonal elements of X, and $\mathrm{Diag}(d)$ is a diagonal matrix in $\mathcal{S}^n$, with the components of d. We define the trace of a matrix $A \in \mathcal{S}^n$ as

$$\mathrm{trace}(A) = \sum_{i=1}^{n} A_{ii} \tag{1.5}$$

and the dot product of two matrices $A, B \in \mathcal{S}^n$ as

$$\begin{aligned} A \bullet B &= \mathrm{trace}(AB) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}B_{ij} \end{aligned} \tag{1.6}$$

We list two useful properties of the trace that shall be useful later. For the second property λ_i, $i = 1, \ldots, n$ denote the n eigenvalues of $A \in \mathcal{S}^n$.

$$\begin{aligned} \mathrm{tr}(AB) &= \mathrm{tr}(BA) \\ \mathrm{tr}(A) &= \textstyle\sum_{i=1}^{n} \lambda_i \end{aligned} \tag{1.7}$$

The first property enables one to rewrite a quadratic function in $\mathcal{R}^n$ as a linear function in a matrix variable $X \in \mathcal{S}^n$.

$$\begin{aligned} x^T A x &= \mathrm{tr}(x^T A x) \\ &= \mathrm{tr}(A x x^T) \\ &= A \bullet X \text{ for } X = x x^T \in \mathcal{S}^n \end{aligned}$$

The first inequality follows since $x^T A x$ is a scalar. This will be useful when we consider semidefinite relaxations of quadratic optimization problems in the later sections. The norm associated with this inner product is the Frobenius norm defined as

$$||A||_F = \sqrt{A \bullet A} \tag{1.8}$$

In the succeeding sections we use trace(X) and tr(X) interchangeably, to denote the trace of the symmetric matrix X.

The following are various ways of characterizing positive semidefinite matrices. Incidentally, these lead to different formulations of $X \succeq 0$ constraint.

1. $X \succeq 0 \Leftrightarrow \lambda_{min}(X) \geq 0$. This characterization uses the smallest eigenvalue $\lambda_{min}(X)$ of X. As we shall see in the next chapter, this is a convex but non-smooth representation. Helmberg and Rendl [52] utilize this idea in their spectral bundle approach.

2. $X \succeq 0 \Leftrightarrow \lambda_i \geq 0, i = 1, \ldots, n$. This characterization uses all the eigenvalues of the symmetric matrix. We shall utilize this formulation to generate valid inequalities in our cutting plane approach in chapter 3.

3. $X \succeq 0 \Leftrightarrow d^T X d \geq 0, \forall d \in \mathcal{R}^n, ||d|| = 1$. This characterization leads to a

semi-infinite LP formulation of the SDP. We shall discuss more details of this formulation in chapter 2.

4. $X \succeq 0 \Leftrightarrow \det(X_{\Sigma,\Sigma}) \geq 0, \forall \Sigma \subset \{1,\ldots,n\}$. This formulation is based on a finite (but exponential) number of smooth constraints, requiring all the principal sub-determinants of a psd matrix to be non-negative.

5. $X \succeq 0 \Leftrightarrow X = V^T V$ for some $V \in \mathcal{R}^{r \times n}$ (r here is the rank of X), and is equivalent to writing the psd matrix as a quadratic equality constraint. This characterization is based on the square-root formulation of the psd constraint. We shall see that Burer et al [20] employ this idea to develop large scale approaches to the SDP (in fact they finally have a non-convex, but infinitely smooth formulation of the SDP).

6. Every psd matrix X has a factorization $X = L\text{Diag}(d)L^T$ (with $d \in \mathcal{R}^n$ unique, and L lower triangular matrix in $\mathcal{M}^n$) (see Horn and Johnson [58] and Golub and Van Loan [39]). Thus we also have

$$X \succeq 0 \Leftrightarrow d_j(X) \geq 0, j = 1,\ldots,n$$

Vanderbei and Benson [117] show the constraints $d_j(X) \geq 0, \forall i$ are concave. Moreover they are also twice continuously differentiable on ${\mathcal{S}_{++}}^n$. Thus we have a convex and smooth formulation of the SDP.

7. Finally, we discuss the *Schur complement* characterization of psd matrices, which is also very useful at times

Theorem 1 *Suppose* $U = \begin{pmatrix} A & B \\ B^T & C \end{pmatrix}$, *with* A *and* C *symmetric, and* $A \succ 0$. *Then*

$$U \succeq 0 \;\; \Leftrightarrow \;\; C - B^T A^{-1} B \succeq 0$$

The matrix $C - B^T A^{-1} B$ is called the *Schur complement* of A in U. A proof can be found in Horn and Johnson [58].

We now have the tools to present an important theorem on psd matrices.

Theorem 2 *A matrix A is psd if and only if*

$$A \bullet B \geq 0, \quad \forall B \in {\mathcal{S}_+}^n$$

Proof: First let us show $A, B \in {\mathcal{S}_+}^n \Rightarrow A \bullet B \geq 0$. Let us assume that A has rank r. Then we have an eigenvalue decomposition $A = P\Lambda_A P^T$, where $P \in R^{n \times r}$, with $P^T P = I_r$. Also $\Lambda_A \in {\mathcal{S}_{++}}^r$. We then have

$$\begin{aligned} A \bullet B &= \operatorname{tr}(P\Lambda_A P^T B) \\ &= \operatorname{tr}(\Lambda_A P^T B P) \\ &= \textstyle\sum_{i=1}^{r} \lambda_i(A) P(:,i)^T B P(:,i) \\ &\geq 0 \end{aligned}$$

The last inequality follows from the fact that $\lambda_i(A) \geq 0$, $\forall i$ (since A is psd), and $P(:,i)^T B P(:,i) \geq 0$, $\forall i$ (since B is psd). To show the reverse implication, note firstly we can consider any rank one matrix $B = xx^T$, without loss of generality, since this choice is psd, and any psd matrix can be expressed as a non-negative combination of these rank one matrices (consider the spectral factorization of B say). Then

$$\begin{aligned} 0 &\leq A \bullet B \\ &= \operatorname{tr}(Axx^T) \\ &= \operatorname{tr}(x^T Ax) \\ &= x^T Ax \end{aligned}$$

and so A is psd too. □

This also implies the following :

Corollary 1 *The psd cone is self-dual, i.e.* ${\mathcal{S}_+}^n = {\mathcal{S}_+}^{n*}$

The primal-dual interior point algorithms for SDP discussed in section 1.6 exploit the self-dual property of the psd cone. Also, not all cones are self dual; an example is the copositive cone discussed in de Klerk and Paschenik [30].

Theorem 3 *The positive semi-definiteness of a matrix can be checked in* $\frac{n^3}{3}$ *arithmetic operations.*

Proof: Use a Cholesky factorization (see Golub and Van Loan [39]). A QR algorithm which determines all the eigenvalues of a matrix can also be implemented in $O(n^3)$ time (though somewhat slower). □

1.3.2 The cone of semidefinite matrices

Theorem 4 *${\mathcal{S}_+}^n$ is a full dimensional, closed convex cone in $\mathcal{R}^{\binom{n+1}{2}}$.*

The set of positive definite matrices ${\mathcal{S}_{++}}^n$ is not a cone because $0 \notin {\mathcal{S}_{++}}^n$. In fact ${\mathcal{S}_{++}}^n$ is the interior of the cone ${\mathcal{S}_+}^n$ and the boundary of ${\mathcal{S}_+}^n$ consists of the positive semidefinite matrices having at least one zero eigenvalue.

It is also instructive to study the faces and extreme points the psd cone (Barker and Carlson [11]); as we shall see in chapter 3 we shall utilize these faces as cutting planes in an interior point approach to solving the SDP. We first begin with the definition of a face.

Definition 2 *A convex set $F \subseteq C$ is called a face of a convex set C if any two elements, $x, y \in C$ with $\alpha x + (1-\alpha)y \in F$, for some $\alpha \in (0,1)$ implies that $x, y \in F$.*

The extreme point is a special face of dimension zero. We can also formally define it as follows

Definition 3 *A point $x \in C$ is called an extreme point of C, if there are no two different points x_1 and x_2 in C such that $x = \frac{1}{2}(x_1 + x_2)$.*

Theorem 5 *The faces of ${\mathcal{S}_+}^n$ are*

1. *The zero matrix $0 \in \mathcal{S}^n$.*
2. *The entire psd cone ${\mathcal{S}_+}^n$.*
3. *They are generated by an orthonormal matrix $P \in \mathcal{R}^{n \times r}$ of rank r, and are of the form*

$$F = \{X : X = PWP^T : W \in {\mathcal{S}_+}^r\} \tag{1.9}$$

Proof: Firstly 0 is a face of ${\mathcal{S}_+}^n$, since it can be expressed only as a convex combination of itself, i.e. $0 = \alpha 0 + (1-\alpha)0$. Also, it is an extreme point by definition 3. The entire psd cone ${\mathcal{S}_+}^n$ is a face too; it is a convex combination of 0 and itself. Now that we have excluded these two cases, consider $F = \{X : X = PWP^T : W \in {\mathcal{S}_+}^r\}$, with $r \in \{1, \ldots, n-1\}$. Thus W is now a psd matrix. Assume, there exists $X \in F$, $X = \alpha A + (1-\alpha)B$, with $A, B \in {\mathcal{S}_+}^n$, not both in F (assume that $A \notin F$ wlog). Then there is a $y \in \mathcal{R}^n$, with $y^T A y > 0$ and $P^T y = 0$. To see this consider an orthogonal basis consisting of the columns of $R = [P \;\; Q]$ for $\mathcal{R}^n$. Here $P \in \mathcal{R}^{n \times r}$ and $Q \in \mathcal{R}^{n \times (n-r)}$ with $P^T P = I_r$ and $Q^T Q = I_{n-r}$ respectively. Furthermore, we can write ${\mathcal{S}_+}^n$ as

$$ {\mathcal{S}_+}^n = \{[P \;\; Q] \begin{bmatrix} W & V \\ V^T & U \end{bmatrix} \begin{bmatrix} P^T \\ Q^T \end{bmatrix} : \begin{bmatrix} W & V \\ V^T & U \end{bmatrix} \succeq 0\} $$

This expression follows from the fact that for the orthogonal matrix R, the linear transformation $X \Rightarrow RXR^T$, maps the set ${\mathcal{S}_+}^n$ onto itself. Since $A \notin F$, we have

$$ A = [P \;\; Q] \begin{bmatrix} W_A & V_A \\ V_A^T & U_A \end{bmatrix} \begin{bmatrix} P^T \\ Q^T \end{bmatrix} $$

with $U_A \neq 0$. Choosing y to be any column of Q, with $(U_A)_{ii} \neq 0$ satisfies $y^T A y > 0$ (since A is psd), and $P^T y = 0$. We then have

$$ \begin{aligned} 0 &= y^T X y \\ &= y^T(\alpha A + (1-\alpha)B)y \\ &> 0 \end{aligned} $$

a contradiction. Here the first equality follows from $P^T y = 0$, and the last inequality follows from $y^T A y > 0$ and $y^T B y \geq 0$ (since B is psd).

We now show that all the faces can be obtained in this way. We first note that 0 is the only face consisting of a single point (extreme point). Now consider any face F that is not an extreme point. For any face F of ${\mathcal{S}_+}^n$, there exists a supporting hyperplane H, with $F = {\mathcal{S}_+}^n \cap H$.

Also, $0 \in F$ for this face F, because any $X \in F$ can be expressed as the convex combination $X = (1-\alpha)0 + \alpha X$, with $\alpha = 1$. Thus we can assume $0 \in H$ as well. So we have

$$F \;=\; \{X : X \succeq 0, W \bullet X = 0\}$$

where we assume $W \in \mathcal{S}^n$ without loss of generality, since $X \in {\mathcal{S}_+}^n$. Consider the spectral decomposition $W = Q\Lambda Q^T$, for $Q \in \mathcal{R}^{n\times(n-r)}$, and $Q^TQ = I_{n-r}$. Since $W \bullet X = 0$, W is singular. Thus we can extend Q to an orthogonal basis $[P, Q]$ for $\mathcal{R}^n$. We have

$$\begin{array}{c} W \bullet X = 0 \\ X \succeq 0 \end{array} \quad \Leftrightarrow \quad \begin{array}{c} X = PWP^T \\ W \succeq 0 \end{array}$$

where $P \in \mathcal{R}^{n\times r}$, and $P^TP = I_r$. □

From (1.2), we have $X = \sum_{i=1}^n \lambda_i x_i x_i^T$ for any $X \in {\mathcal{S}_+}^n$. In addition $\lambda_i \geq 0$, $i = 1, \ldots, n$. From Theorem 5 it follows that $\{\lambda xx^T : \lambda \geq 0\}$ is a face of X. Since these faces cannot be expressed as convex combination of smaller faces, they form a minimal generating set for ${\mathcal{S}_+}^n$. Moreover these faces $\{X : X = xx^T, ||x|| = 1\}$ cannot be generated from a finite set, with the result that the SDP cone is no longer *polyhedral* unlike the LP cone. Also since the faces of ${\mathcal{S}_+}^k$ have dimension $\frac{k(k+1)}{2}$, there is a considerable jump in dimension in going from on face to the next. Some of the dimensions are not attained, again this is in sharp contrast to polyhedral cones.

1.4 Semidefinite Programming

Consider the following pair of semidefinite programming problems

$$\begin{array}{lrcll} \min & C \bullet X & & & \\ \text{s.t.} & A_i \bullet X & = & b_i, \;\; i = 1, \ldots, k & \qquad (SDP) \\ & X & \succeq & 0 & \end{array}$$

with dual

$$\begin{array}{lrcll} \max & b^Ty & & & \\ \text{s.t.} & \sum_{i=1}^k y_i A_i + S & = & C & \qquad (SDD) \\ & S & \succeq & 0 & \end{array}$$

where the variables $X, S \in \mathcal{S}^n$, the space of real symmetric $n \times n$ matrices, $b \in \mathcal{R}^k$, the matrices $A_i, i = 1, \ldots, k \in \mathcal{S}^n$ and $C \in \mathcal{S}^n$ are the given problem parameters. The constraints $X \succeq 0$, $S \succeq 0$ are the only nonlinear (actually convex) constraints in the problem requiring that these matrices X and S be psd. We will assume hereafter that the matrices A_i, $i = 1, \ldots, k$ are linearly independent, i.e. $k \leq \binom{n+1}{2}$.

Note we can assume C and A_i, $i = 1, \ldots, k$ to be symmetric wlog. If they were not, then we could decompose C (say) into its symmetric and skew symmetric components (a matrix A is skew symmetric if $A = -A^T$) C_{symm} and C_{skew} respectively. Also, $X \in \mathcal{S}^n$; it is then easy to show that $C_{skew} \bullet X = 0$, and thus $C \bullet X = C_{symm} \bullet X$.

We can think of the SDP as a generalization of the standard linear programming problem.

$$\begin{array}{lrcll} \min & c^T x & & & \\ \text{s.t.} & a_i^T x & = & b_i, \quad i = 1, \ldots, k & \qquad (LPP) \\ & x & \geq & 0 & \end{array}$$

with dual

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & \sum_{i=1}^k y_i a_i + s & = & c & \qquad (LPD) \\ & s & \geq & 0 & \end{array}$$

Here a_i denote the rows of a $k \times n$ matrix A. If we compare the two, we find that the SDP is indeed an LP, where the vectors c, x, s, a_i's are replaced by matrices C, X, S, A_i's respectively and the element-wise non-negativity constraints $x \geq 0, s \geq 0$ are replaced by the requirement that $X \succeq 0, S \succeq 0$. Thus semidefinite programming is essentially linear programming over the cone of semidefinite matrices. Henceforth, we shall denote the constraints in (SDP) and (SDD) as

$$\begin{aligned} \mathcal{A}(X) &= b \\ \mathcal{A}^T y + S &= C \end{aligned}$$

respectively. Here $\mathcal{A} : \mathcal{S}^n \rightarrow R^k$ and $\mathcal{A}^T : R^k \rightarrow \mathcal{S}^n$ are of the form

$$\mathcal{A}(X) \;=\; \begin{bmatrix} A_1 \bullet X \\ \vdots \\ A_k \bullet X \end{bmatrix} \quad \text{and} \quad \mathcal{A}^T y \;=\; \sum_{i=1}^k y_i A_i$$

with $A_i \in \mathcal{S}^n, i = 1, \ldots, k$. $C \in \mathcal{S}^n$ is the cost matrix, $b \in R^k$ the RHS vector.

Theorem 6 *(Weak Duality) If X is feasible in (SDP) and (y,S) in (SDD), then*

$$C \bullet X - b^T y \;=\; X \bullet S \;\geq\; 0 \tag{1.10}$$

Proof:

$$\begin{aligned} C \bullet X - b^T y &= (\textstyle\sum_{i=1}^k y_i A_i + S) \bullet X - b^T y \\ &= \textstyle\sum_{i=1}^k (A_i \bullet X) y_i + S \bullet X - b^T y \\ &= S \bullet X \end{aligned}$$

Since X and S are semidefinite matrices we have $S \bullet X = X \bullet S \geq 0$ (see Theorem 2).

□

We can actually derive (SDD) as the Lagrangian dual of (SDP). To see this, we introduce the linear equalities $A_i \bullet X = b_i, i = 1, \ldots, k$ into the objective function by means of Lagrangian multipliers y_i, $i = 1, \ldots, k$. The Lagrangian dual is then

$$\begin{aligned} \max_{y \in \mathcal{R}^k} \{ \min_{X \succeq 0} C \bullet X - \textstyle\sum_{i=1}^k y_i (A_i \bullet X - b_i) \} &= \\ \max_{y \in \mathcal{R}^k} \{ b^T y + \min_{X \succeq 0} (C - \textstyle\sum_{i=1}^k y_i A_i) \bullet X \} &= \\ \max_{y \in \mathcal{R}^k, S = (C - \mathcal{A}^T y) \succeq 0} b^T y & \end{aligned}$$

The last equality follows from Theorem 2, and so the quantity in the inner minimization is zero. Also, the aim of the outer maximization in the Lagrangian relaxation is to find the best lower bound for (SDP).

We call the difference between the optimal value of (SDP) and (SDD), which is always non-negative from Theorem 6 the *duality gap*. Strong duality on the other hand, is the assertion that the duality gap is zero and that both problems attain their optima whenever both problems are feasible. This is true for the LP, but it

does not always hold for SDP problems. We shall return to the notion of *strong duality* and conditions under which it holds in Chapter 2 (see section 2.3).

1.5 Examples

In this section, we present three applications of semidefinite programming. These applications are taken from Vandenberghe and Boyd [115]. The first example in section 1.5.1 appears in control theory, while the SDP relaxation appearing in section 1.5.2 was used by Goemans and Williamson [36] in their celebrated 0.878 approximation algorithm for the max cut problem. The third example in section 1.5.3 is on the minimum eigenvalue of a symmetric matrix.

1.5.1 Example 1 : Minimizing the maximum eigenvalue

Problems of this type arise in control theory, structural optimization, combinatorial optimization etc. For example in control theory it might appear as stabilizing a differential equation.

The problem is the following : Suppose we have a symmetric matrix, say $A(x)$ depending affinely on a vector $x \in \mathcal{R}^k$: $A(x) = A_0 + \sum_{i=1}^{k} x_i A_i$, where $A_i \in \mathcal{S}^n, i = 1, \ldots, k$. Now we wish to choose x to minimize the maximum eigenvalue of $A(x)$ denoted by $\lambda_{max}(A(x))$.

Note that $\lambda_{max}(A(x)) \leq t$ iff $\lambda_{max}(A(x) - tI) \leq 0$. Since $\lambda_{max}(.)$ is a convex function we have $\lambda_{max}(A(x) - tI) = -\lambda_{min}(tI - A(x))$. Thus we have $\lambda_{min}(tI - A(x)) \geq 0$. This holds iff $tI - A(x) \succeq 0$. The SDP in dual form (SDD) is

$$\begin{array}{ll} \max & -t \\ \text{s.t.} & tI - A(x) \ \succeq \ 0 \end{array} \tag{1.11}$$

The dual variable is $y = (t, x)$.

1.5.2 Example 2 : The Max Cut Problem

Given a graph $G = (V, E)$, and a non-negative weight vector $w = w_{ij} \in {\mathcal{R}_+}^E$. For $S \subseteq V$, $\delta(S)$ denotes $\delta(S) = \{(i, j) \in E : i \in S, j \notin S\}$, the *cut* determined by S,

with *weight* equal to $w(\delta(S)) = \sum_{i,j\in\delta(S)} w_{ij}$. We want to find the cut of maximum weight.

We can assume that the graph is complete by setting $w_{ij} = 0$ for all non edges (i, j); we also set $w_{ii} = 0$, $\forall i$.

Let us define a vector $x \in \mathcal{R}^n$, with $x_i = 1$, if $i \in S$ and -1 otherwise. Then $x_i x_j = -1$ if $(i, j) \in \delta(S)$ and 1 otherwise. Now define the Laplacian matrix $L \in \mathcal{S}^n$, by setting $L_{ij} = -\frac{w_{ij}}{4}$, $i \neq j$, and $L_{ii} = \sum_j \frac{w_{ij}}{4}$, $\forall i$.

We then have

$$\begin{aligned} w(\delta(S)) &= \tfrac{1}{2}\sum_{i<j} w_{ij}(1 - x_i x_j) \\ &= \tfrac{1}{4}\sum_i \sum_j w_{ij}(1 - x_i x_j) \\ &= x^T L x \end{aligned}$$

Since every $(1, -1)$ vector corresponds to a cut, the max cut problem is equivalent to the following integer quadratic programming problem

$$\max \quad x^T L x, \quad x_i \in \{1, -1\}, i \in V$$

This is also equivalent to the following non-convex quadratically constrained quadratic problem

$$\max \quad x^T L x, \quad x_i^2 = 1, i \in V \tag{1.12}$$

Let us define the rank one matrix $X = xx^T$. We have $X_{ii} = x_i^2 = 1, \forall i$ and $X_{ij} = x_i x_j$. Also $X \succeq 0$, since $d^T X d = (d^T x)^2 \geq 0$. Now we rewrite the objective function in (1.12) as

$$\begin{aligned} x^T L x &= \mathrm{tr}(x^T L x) \\ &= L \bullet xx^T \\ &= L \bullet X \end{aligned}$$

The first equality follows from the fact that $x^T L x$ is a scalar, and the second follows from $\mathrm{tr}(AB) = \mathrm{tr}(BA)$.

Thus (1.12) is equivalent to

$$\begin{array}{lrcl} \max & L \bullet X & & \\ \text{s.t.} & X_{ii} & = & 1, \quad i = 1, \ldots, n \\ & X & \succeq & 0, \quad X \text{ rank one} \end{array}$$

If we relax the last constraint, we get the SDP relaxation (1.13) of the max cut problem

$$\begin{array}{lrcl} \max & L \bullet X & & \\ \text{s.t.} & X_{ii} & = & 1, \quad i = 1, \ldots, n \\ & X & \succeq & 0 \end{array} \tag{1.13}$$

1.5.3 Example 3 : The minimum eigenvalue of a symmetric matrix

As a final example, we express the minimum eigenvalue of a symmetric matrix as an SDP. This characterization of the minimum eigenvalue will also be useful, when we motivate the spectral bundle method in section 1.7.1 and Chapter 2.

From the variational characterization of the minimum eigenvalue of a symmetric matrix A (Horn and Johnson [58]), we have

$$\begin{array}{rcl} \lambda_{min}(A) & = & \min \quad x^T A x \\ & & x^T x = 1 \end{array} \tag{1.14}$$

Now define a matrix $X = xx^T$. The objective function in (1.14) can be expressed as $x^T A x = \text{tr}(Axx^T) = A \bullet X$.

Also $x^T x = \text{tr}(xx^T) = \text{tr}(X) = 1$. Finally, $X = xx^T$ implies that X is rank one and positive semidefinite.

A SDP relaxation of the minimum eigenvalue is

$$\begin{array}{rcl} \lambda_{min}(A) & = & \min \quad A \bullet X \\ & & \{X : \text{tr}(X) = 1, X \succeq 0\} \end{array} \tag{1.15}$$

In proceeding from (1.14) to (1.15), we have dropped the requirement that X be rank one. However this is not necessary for the following reason (Overton [95]).

Theorem 7 *The convex hull of the set*

$$\{vv^T : v \in \mathcal{R}^n, v^T v = 1\}$$

is the set

$$\{X : X \in \mathcal{S}^n : tr(X) = 1, X \succeq 0\}$$

Furthermore, the elements in the first set are the extreme points of the second set.

Proof: Any convex combination of the first set is clearly contained in the second. To see this, consider a convex combination $X = \sum_i \lambda_i v_i v_i^T$. Since $\lambda_i \geq 0$, and the $v_i v_i^T$ are rank one psd matrices, the matrix X is clearly psd too. Also

$$\begin{aligned} \mathrm{tr}(X) &= \mathrm{tr}(\textstyle\sum_i \lambda_i v_i v_i^T) \\ &= \textstyle\sum_i \lambda_i v_i^T v_i \\ &= \textstyle\sum_i \lambda_i \\ &= 1 \end{aligned}$$

Furthermore, any matrix in the second set has a spectral decomposition

$$X = \textstyle\sum_{i=1}^n \lambda_i v_i v_i^T$$

Since X is psd, we have $\lambda_i \geq 0$, $i = 1, \ldots, n$. Also $\mathrm{tr}(X) = 1$ implies that $\sum_{i=1}^n \lambda_i = 1$, and we assume that the eigenvectors v_i are normalized to have unit norm. Thus the right hand side is a convex combination of n elements in the first set and we are done. Clearly, any element of the first set is an extreme point of the second. Also, any element of the second set which is not rank one can be written as a nontrivial convex combination of elements in the first set, and is no longer an extreme point.

□

Now $C \bullet X$ is a linear function in the matrix variable X, and minimizing a linear function over a set is the same as minimizing it over its convex hull. Thus (1.14) and (1.15) are equivalent.

We sketch the feasible region of the SDP (1.15) in figure 1.1. The figure was

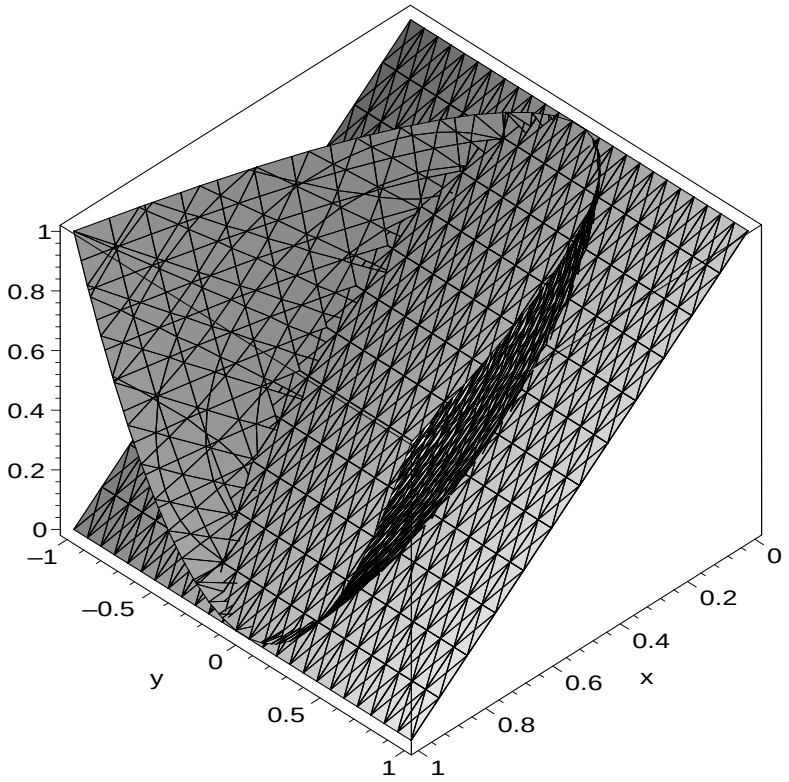


Figure 1.1: Feasible region of minimum eigenvalue SDP (1.15) for a 2×2 symmetric matrix

obtained by setting $X = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$, and plotting the region $\det(X) = 0, \text{Trace}(X) = 1, 0 \leq \begin{bmatrix} x \\ z \end{bmatrix} \leq 1$, and $\frac{1}{\sqrt{2}} \leq y \leq \frac{1}{\sqrt{2}}$. The extreme points are the points of intersection of any rank one matrix X with $y^2 = xz$, and the plane $x + z = 1$ $(\text{tr}(X) = 1)$.

1.6 Interior Point Algorithms for SDP

Interior Point methods for SDP problems were first proposed by Alizadeh [1], and Nesterov and Nemirovskii [90]. Alizadeh [1] showed that most, if not all primal or dual algorithms for linear programming could be extended to SDP in a mechanical way. On the other hand Nesterov and Nemirovskii [90] presented a deep and unified theory of interior point methods for solving general cone programming

problems, using the notion of self concordant barrier functions. In particular, since the cone of positive semidefinite matrices admits a self concordant barrier function, they showed that semidefinite programs can be *solved in polynomial time*. Recently Schmieta and Alizadeh [106] have developed a general framework, whereby analysis of interior point methods for SDP can also be extended to other classes of symmetric cones.

The basic idea is to transform a constrained optimization problem (in this case a SDP) into an unconstrained problem. One way to handle inequality constraints consists in adding a barrier term to the cost function. In case of minimization problems, the barrier term is small in the interior of the feasible region, but grows to infinity as we approach the boundary of the feasible region. Thus if we start in the interior of the feasible region, the barrier term prevents us from leaving the feasible region. Furthermore, a descent direction in a point close to the boundary will automatically point away from the boundary. However the optimal solution is usually located on the boundary. In order to produce a sequence of iterates that converge to the optimum, a mechanism has to be provided that reduces the influence of the barrier function as the optimization process continues. This is achieved by weighting the barrier term and diminishing the weight successively. This is a sequential unconstrained minimization technique, and under certain conditions the minima of the sequence of barrier problems can be shown to converge to an optimal solution of the original problem. Typically the minima of the subproblems is not computed exactly but approximated by a few Newton steps. Since Newton's method exploits second order information, the algorithms converge very fast. An approximate optimal solution (to any degree of precision) is obtained within a polynomial number of iterations. On the other hand the computation of a single step is computationally rather expensive and this limits these methods to problems of moderate size, say around 3000 constraints.

The important references on interior point methods for SDP include Alizadeh et al [3], Helmberg et al [54], Kojima et al [68], Monteiro [87], Todd et al [112]. These are primal dual algorithms. Benson et al [12] propose a dual scaling algorithm. Finally, Zhang [122] develops a framework to extend some primal dual interior point

algorithms from linear programming to semidefinite programming. A good overview also appears in Etienne de Klerk's Ph.D. thesis [29].

We wish to emphasize the following points, before we begin our discussion of interior point methods for the SDP.

1. It was fairly easy to extend primal only and dual only interior point methods from the LP to the SDP case (see Alizadeh [1] and Nesterov and Nemirovskii [90]). In fact, a *word by word* case extension is possible.

2. However this is not so with primal-dual interior point algorithms for the LP; the most popular interior point algorithms for LP from a computational perspective. The reason is the following : We cannot directly apply Newton's method to (1.16), since after linearizing we do not have a square system of equations in (1.17). Since

$$X, S \in \mathcal{S}^n \quad \not\Rightarrow \quad XS \in \mathcal{S}^n$$

we have $k + n^2 + \frac{n(n+1)}{2}$ equations, and only $k + n(n+1)$ unknowns in (1.17). (We are using the fact that XS is not symmetric, but any matrix in $\mathcal{M}^n$ giving us n^2 equations in the process). Thus, we have an *overdetermined* system of linear equations. This is unlike the LP case, where X and S are diagonal matrices and hence commute. There are several ways to overcome this, giving us more than one primal dual interior point algorithm for the SDP. We shall return to this later in the section.

In this section we discuss interior point algorithms for the SDP. We present a generic interior point algorithm for the SDP and consider the computational issues involved.

Consider the semidefinite programming problem

$$\begin{array}{lrcll} \min & C \bullet X & & & \\ \text{subject to} & \mathcal{A}(X) & = & b & (SDP) \\ & X & \succeq & 0, & \end{array}$$

with dual

$$\begin{array}{lllllll} \max & & b^T y & & & & \\ \text{subject to} & \mathcal{A}^T y & + & S & = & C & \qquad (SDD) \\ & & & S & \succeq & 0 & \end{array}$$

where $X, S \in \mathcal{S}^n$, the space of real symmetric $n \times n$ matrices.

We will assume that strong duality holds for this pair, i.e. both the primal as well as the dual are strictly feasible (we shall discuss this notion in more detail in section 2.3).

Interior point algorithms start within the cone of positive semidefinite matrices. In order to avoid leaving the cone during the optimization process, the problem of minimizing the original SDP is replaced by the problem of approximately minimizing a sequence of auxiliary barrier problems. The auxiliary barrier problem is of the form

$$\begin{array}{lll} \min & C \bullet X - \mu \log\det(X) & \\ \text{s.t.} & \mathcal{A}(X) & = b \\ & X & \succ 0 \end{array}$$

Here $\mu > 0$ is the barrier parameter and $-\log\det(X)$ is the *self concordant* barrier function. This barrier function grows to infinity if an eigenvalue of X tends to zero i.e. as we approach the boundary of the semidefinite cone.

We transform the barrier problem into an unconstrained problem, by introducing a Lagrange multiplier y for the equality constraints.

$$L_\mu(X, y) \;=\; C \bullet X - \mu \log\det(X) + y^T(b - \mathcal{A}(X))$$

Note that $L_\mu(X, y)$ is a strictly convex function on ${\mathcal{S}_+}^n$. Writing down the *KKT* conditions for optimality we have.

$$\begin{array}{lllll} \bigtriangledown_X L_\mu & = & C - \mu X^{-1} - \mathcal{A}^T y & = & 0 \\ \bigtriangledown_y L_\mu & = & b - \mathcal{A}(X) & = & 0 \end{array}$$

Setting $S = \mu X^{-1}$ we can rewrite these conditions as

$$\begin{aligned} \mathcal{A}(X) &= b, \quad X \succ 0 \\ \mathcal{A}^T y + S &= C, \quad S \succ 0 \\ XS &= \mu I \end{aligned} \tag{1.16}$$

The first two conditions correspond to primal and dual feasibility. For $\mu = 0$, the third condition corresponds to the complementary slackness condition $XS = 0$.

We denote the solution to (1.16) for some fixed $\mu > 0$ by (X_μ, y_μ, S_μ). This forms the *central path* which is a smooth curve, and which converges to the optimal solution (X^*, y^*, S^*), as $\mu \to 0$.

If we solve (1.16) by Newton's method, we get the following linearized system

$$\begin{aligned} \mathcal{A} \triangle X &= -(\mathcal{A}X - b) &= F_p \\ \mathcal{A}^T \triangle y + \triangle S &= -(\mathcal{A}^T y + S - C) &= F_d \\ \triangle XS + X \triangle S &= \mu I - XS \end{aligned} \tag{1.17}$$

Since X and S are matrices, they do not always commute i.e. $XS \neq SX$. In general we cannot expect that there exist symmetric $\triangle X$ and $\triangle Z$ that solve (1.17). However we can introduce a symmetrization operator

$$H_P(M) = \tfrac{1}{2}(PMP^{-1} + (PMP^{-1})^T) \tag{1.18}$$

In the simplest case when $P = I$ we get $H_I(M) = \frac{1}{2}(M + M^T)$. Different P correspond to different search directions. A good survey of the various search directions appears in Todd [111]. We are now ready to present the algorithmic framework for a typical interior point scheme for the SDP.

Generic Interior Point Algorithm for the SDP

Input : $\mathcal{A}, b, C$ and some starting point (X^0, y^0, S^0) with $X^0 \succ 0$ and $S^0 \succ 0$.

1. Choose the barrier parameter μ.
2. Compute $(\triangle X, \triangle y, \triangle S)$ by solving (1.17). Modify ΔX which need not be symmetric using (1.18) for a specific choice P. For the ease of exposition we

shall consider the simplest case where $P = I$. This incidentally gives the Alizadeh-Haeberly-Overton (AHO) direction (Alizadeh et al [3]).

3. Choose some $\alpha \in (0, 1]$ so that $X + \alpha \triangle X$ and $S + \alpha \triangle S$ are positive definite.

4. Set $(X, y, S) = (X + \alpha \triangle X, y + \alpha \triangle y, S + \alpha \triangle S)$.

5. If $||\mathcal{A}X - b||$ and $||\mathcal{A}^T y + S - C||$ and $X \bullet S$ are small enough, then **stop**, else **goto 1**.

We need strong restrictions on the initial starting point, and the values of μ and α to prove polynomial iteration complexity. The important point is that we can solve an SDP to a tolerance ϵ in $O(\sqrt{n}\log(\frac{1}{\epsilon}))$ such iterations (see Todd [110] for more details). This complexity bound is for short step primal dual interior point algorithm. Interestingly, this is same bound as in the LP case.

We now examine the work involved in each iteration. Consider (1.17). We first express $\triangle X$ in terms of $\triangle S$, and then write $\triangle S$ in terms of $\triangle y$. This yields the following system depending on $\triangle y$ only.

$$\begin{aligned} \mathcal{A}(X\mathcal{A}^T(\triangle y)S^{-1}) &= b - \mathcal{A}(\mu S^{-1} + XF_dS^{-1}) \\ \triangle S &= F_d - \mathcal{A}^T \triangle y \\ \triangle X &= \mu S^{-1} - X - X \triangle S S^{-1} \end{aligned} \tag{1.19}$$

After solving this system, we symmetrize $\triangle X$ by setting

$$\triangle X = \tfrac{1}{2}(\triangle X + \triangle X^T)$$

Consider $My = \mathcal{A}(X\mathcal{A}^T(y)S^{-1})$. The ith row of My is given by

$$A_i \bullet X\mathcal{A}^T(y)S^{-1} = \textstyle\sum_{j=1}^{k} y_j \text{Trace}(XA_iS^{-1}A_j)$$

Therefore we have $M_{ij} = \text{Trace}(XA_iS^{-1}A_j)$. This matrix is symmetric and positive definite, if we assume $\mathcal{A}(.)$ to have full row rank.

Solving for $\triangle y$ requires $\frac{k^3}{3}$ arithmetic operations, using a Cholesky decomposition. Moreover M has be to recomputed in each iteration. An efficient way to

build one row of M is the following

1. Compute XA_iS^{-1} once in $O(n^3)$ time
2. Determine the k single elements via $XA_iS^{-1} \bullet A_j$ in $O(kn^2)$.

In total the construction of M requires $O(kn^3 + k^2n^2)$ arithmetic operations and so the construction of M is the most expensive operation in each iteration.

In many applications the constraint matrices A_i have a special structure. For most of the combinatorial optimization problems such as max cut etc, these matrices have a rank one structure. This reduces the computation time to $O(kn^2 + k^2n)$ operations. Benson et al [12] have proposed a dual scaling algorithm that exploits this rank one structure, and the sparsity in the dual slack matrix. However in their approach the matrix M is dense, and the necessity to store and factorize this dense matrix M limits the applicability of these methods to problems with around 3000 constraints on a well equipped work station.

A way to overcome the problem of having to store the matrix M via the use of an iterative scheme, which only accesses this matrix through matrix vector multiplications, is discussed in Toh and Kojima [113]. This approach is not straightforward though : the Schur matrix M becomes increasingly ill-conditioned as the iterates approach the boundary. Hence there is a need for good pre-conditioners, if the iterative method is to converge quickly.

Another drawback of primal-dual interior point methods is that the matrix X is completely dense, regardless of the sparsity of the problem. In this regard, Fukuda et al [34] employ ideas from the completion of psd matrices (Grone et al [41], Laurent [73]) to resolve this issue.

1.7 Large Scale Approaches

As seen in the previous section, interior point methods are fairly limited in the size of problems they can handle. We now turn our focus to methods that can handle large scale SDP's. Most of these methods place special restrictions on the SDP problems they handle. Also, we only have a proof of global convergence of the method, no longer polynomial complexity. It is also interesting to note that most of

these methods attempt to solve the dual (SDD), thereby exploiting sparsity of large scale problems. We describe two of the more successful approaches in this section.

1. The spectral bundle method due to Helmberg and Rendl [52]. This method recasts (SDD) as an eigenvalue optimization problem, which is *convex but non-smooth*, and solves this problem using bundle methods for non-differentiable optimization. An additional assumption is that $\mathrm{tr}(X) = a$, i.e. the trace of every feasible X is constant. We discuss this approach in section 1.7.1.

2. A nonlinear programming formulation due to Burer et al [21]. This scheme recasts (SDD) as a *non-convex but infinitely smooth* problem, and solves it using a first order log barrier approach. An additional assumption is $\mathrm{diag}(X) = d$, i.e. the diagonal entries of the X are fixed. We discuss this in section 1.7.2.

Other large scale approaches include Fukuda et al [33], and Vanderbei and Benson [117].

1.7.1 The Spectral bundle method

We present a brief overview of the spectral bundle method in this section. Since the vectors in the bundle P maintained by this scheme, will be used as constraints in our LP approach in in Chapter 2, we defer a detailed description of this method to that chapter.

The spectral bundle scheme is due to Helmberg and Rendl [52]. The bundle is suitable for large m. However it is only a first order method, with no polynomial bound on the number of iterations, unlike the interior point methods, discussed in the previous section. Nevertheless excellent computational results have been obtained for problems that are inaccessible to the latter methods due to their size. A second order method which converges globally and which enjoys asymptotically a quadratic rate of convergence was recently developed by Oustry [93]. Helmberg and Kiwiel [50] also extended the method to problems with bounds.

The essential idea of the method is the following : Consider the eigenvalue optimization problem

$$\max_y \quad a\lambda_{min}(C - \mathcal{A}^T\mathrm{y}) \quad + \quad b^T\mathrm{y} \tag{1.20}$$

Problems of this form are equivalent to the dual of semidefinite programs (SDP), whose primal feasible set has a constant trace, i.e. $\text{Trace}(X) = a$ for all $X \in \{X \succeq 0 : \mathcal{A}(X) = b\}$. This can be easily verified as follows. From section 1.5.3 on the minimum eigenvalue function, we have $\lambda_{min}(C - \mathcal{A}^T y) = \min_{X:\text{tr}X=1, X\succeq 0}(C - \mathcal{A}^T y) \bullet X$. If this X also satisfies $\mathcal{A}X = b$, then we are essentially solving the primal problem (SDP), with the additional constraint that $\text{tr}(X) = a$. Also we require that either the primal or the dual SDP contain a strictly feasible point, so that strong duality $XS = 0$ holds for the optimal primal dual pair.

In the spectral bundle scheme the maximum eigenvalue is approximated by means of vectors in the subspace spanned by the bundle P. This amounts to solving the following problem (1.21) in lieu of (1.20).

$$\max_y \quad a\lambda_{min}(P^T(C - \mathcal{A}^T y)P) + b^T y \tag{1.21}$$

This in turn is equivalent to solving the following (SDP)

$$\min_{X=PWP^T:\text{tr}(W)=a, \mathcal{A}(PWP^T)=b, W\succeq 0} \quad C \bullet (PWP^T) \tag{1.22}$$

(1.22) implies that we are approximately solving (SDP), by considering only a subset of the feasible X matrices. By keeping the number of columns r in P small, the resulting SDP can be solved quickly. Helmberg and Rendl [52] using results from Pataki [99] were able to show that r is bounded by $\binom{r+1}{2} \leq k+1$, i.e. the dimension of the subspace P is roughly bounded by the square root of number of constraints. The optimum solution of (1.21) typically produces an indefinite dual slack matrix. The negative eigenvalues and corresponding eigenvectors are used to update the subspace, i.e. P and the process is iterated.

To summarize the essential idea of the bundle method is the following :

1. Consider a subset $X = PVP^T$ of the feasible X, where $V \in \mathcal{S}^r$ with $r = O(\sqrt{k})$. (exploit Theorem 16 due to Pataki [99]).

2. Use this to find a good y (the direction finding subproblem discussed in detail in Chapter 2).

3. Use y in turn to improve X (update the bundle; details again discussed in Chapter 2)

1.7.2 Solving a class of SDP's via nonlinear programming

We now turn to the method due to Burer, Monteiro and Zhang [20]. This method complements the bundle approach discussed in the previous section : it recasts the dual SDP as a non-convex but smooth unconstrained problem. We require that the diagonal entries of the primal matrix X be fixed. In fact, the method is also applicable to nonlinear SDP's, but a proof of convergence exists only in the linear case. More details can be found in Burer et al [20], and computational results are discussed in Burer et al [22].

Consider the following SDP and its dual

$$\begin{array}{lrcl} \max & C \bullet X & & \\ \text{s.t.} & \operatorname{diag}(X) & = & d, \\ & \mathcal{A}(X) & = & b, \\ & X \succeq 0 & & \end{array} \tag{1.23}$$

with dual

$$\begin{array}{lrcl} \min & d^T z + b^T y & & \\ & S & = & \operatorname{Diag}(z) + \mathcal{A}^T y - C, \\ & S \succeq 0 & & \end{array} \tag{1.24}$$

Burer et al suggest solving (1.24) by means of an equivalent nonlinear programming problem, obtained by eliminating variables. They consider only strictly feasible solutions of (1.24), i.e. $S \succ 0$. We discuss the main steps in their derivation below. Also, the notation $L \in \mathcal{L}^n$ implies that the matrix L is lower triangular, and $L \in {\mathcal{L}_+}^n ({\mathcal{L}_{++}}^n)$ asserting the diagonal entries of the lower triangular L are non-negative (strictly positive).

1. Every $S \in {\mathcal{S}_{++}}^n$ can be uniquely factored as LL^T, with a positive diagonal in L (this follows from the Cholesky factorization). Decompose $L = \operatorname{Diag}(w) + L_0$,

with $w \in {\mathcal{R}_{++}}^n$, and L_0 strictly lower triangular. Thus we have

$$\text{Diag}(z) + \mathcal{A}^T y - C \quad = \quad (\text{Diag}(w) + L_0)(\text{Diag}(w) + L_0)^T \tag{1.25}$$

There are $n + k + \frac{n(n+1)}{2}$ variables, and $\frac{n(n+1)}{2}$ equations in (1.25).

2. Then, the set

$$\{(z, y, S) \in \mathcal{R}^n \times \mathcal{R}^k \times {\mathcal{S}_{++}}^n \quad : \quad S = \text{Diag}(z) + \mathcal{A}^T y - C\}$$

is in bijective correspondence with

$$\{(z, y, L) \in \mathcal{R}^n \times \mathcal{R}^k \times {\mathcal{L}_{++}}^n \quad : \quad \text{Diag}(z) + \mathcal{A}^T y - C = (\text{Diag}(w) + L_0)(\text{Diag}(w) + L_0)^T$$

3. Use the $\frac{n(n+1)}{2}$ equations in (1.25) to uniquely express $L_0 \in \mathcal{R}^{\frac{n(n-1)}{2}}$ and $z \in \mathcal{R}^n$ in terms of $w \in {\mathcal{R}_{++}}^n$ and $y \in \mathcal{R}^k$ (we have now eliminated the variables L_0 and z).

Carrying out the above steps, we arrive at the following equivalent formulation (1.26) of (1.24)

$$\begin{array}{lll} \inf & d^T z(w, y) + b^T y & \\ \text{s.t.} & w & > \quad 0 \end{array} \tag{1.26}$$

A few points are now in order :

1. The nonlinearity $S \succeq 0$ has been shifted from the constraints to the objective function, i.e. in the term $z(w, y)$.

2. The feasible region for (1.26) is open due to the requirement $w > 0$.

3. The unconstrained problem does not attain its optimal solution, since (1.24) attains its optimal solution only on the boundary, i.e. where $S \succeq 0$. Hence we take the infimum in (1.26). This need not deter us, since we are not interested in solving (1.24) exactly, only to a sufficiently small $\epsilon > 0$.

4. The function $z(w, y)$ is infinitely differentiable, and so the SDP has been transformed into a non-convex but smooth unconstrained problem (this is contrast to the bundle scheme where we have a convex but non-smooth formulation).

The authors then suggest a log barrier method and a potential reduction method to solve (1.26). The main computational task is the computation of the gradient, and Burer et al [21] develop formulas that exploit the sparsity of the problem data. Although the objective function is non-convex, the authors prove global convergence of their method, and have obtained excellent computational results on large scale problems.

1.8 A linear programming (LP) approach to SDP

We have discussed the semidefinite programming problem (SDP) in this chapter. We have also mentioned interior point methods, the traditional ways of solving SDP's in section 1.6. We have discussed the merits and shortcomings of interior point methods in this section; however it is worth reiterating them again.

The advantages of interior point methods are :

1. Interior point methods work on all reasonably posed SDP's. This is in sharp contrast to the large scale approaches we discussed in section 1.7 which place special restrictions on the SDP's handled.

2. Interior point methods can be used to solve an SDP to any degree of precision, in polynomial time. In fact, the worst case complexity is $O(\sqrt{n}\log(\frac{1}{\epsilon}))$ iterations for a short step primal dual algorithm (interestingly this is the same as that for an LP) solving the SDP to an accuracy of $\epsilon > 0$. (Refer to section 1.6 for the work involved in each iteration).

3. Interior point methods exploit second order information : they use Newton's method to solve the sequence of unconstrained subproblems that arise in the approach. Thus the method converges quickly, as opposed to the large scale approaches in section 1.7, which utilize only first order information. As a result, these methods converge very slowly as they approach the optimal solution.

On the other hand, interior point methods have their own shortcomings :

1. Interior point methods are fairly limited in the size of problems they can handle. As mentioned in section 1.6, the main computational task is the computation of the Schur matrix, in computing the search direction. This limits the applicability of these methods to problems with not more than 3000 constraints on a well equipped work station.

2. Our aim in this thesis is to solve combinatorial optimization problems to optimality using a branch and cut approach. From this perspective the more serious shortcoming of interior point methods is that there is no convincing warm start strategy, that allows one to quickly re-optimize a slightly perturbed version of the original SDP after the addition of cutting planes. Again, this is due to the lack of a easily implementable simplex like scheme for the SDP.

The objective of this thesis is linear programming (LP) approaches to semidefinite programming (SDP) problems. This potentially allows one to solve large scale problems approximately and quickly, using state of the art linear solvers available. This linear approach can be incorporated in a branch and cut approach to solving integer programming problems. We must emphasize an important point here : we are not interested in solving the SDP to optimality. This is not needed in our applications since all we care about is generating good bounds on the optimal values of combinatorial optimization problems, to be solved in a cutting plane approach.

The chapters in this thesis are organized as follows :

1. An SDP with a bounded feasible set can be recast as an eigenvalue optimization problem. This is a convex but non-smooth problem, typically tackled by bundle methods for non-differentiable optimization. This is the basis of the spectral bundle method discussed in Chapter 2. We provide evidence in this chapter that only a small number of linear constraints, namely those in the bundle maintained by the spectral bundle approach, bounded by the square root of the number of constraints in the SDP, are typically required in the LP relaxations.

2. The SDP can also be expressed as a *semi-infinite* LP, with a polynomial time separation oracle. (we shall define these terms in Chapter 3). Thus in practice it can be solved using a cutting plane LP approach; we present an approach in Chapter 3. However to make the resulting method competitive several refinements are necessary; we discuss these in detail in Chapter 3.

3. We incorporate this interior point cutting plane LP approach to the SDP in an overall cutting plane approach for integer programming problems. This is the subject of Chapter 4; in particular we discuss the approach on the maxcut problem.

4. Finally, we discuss conclusions, foreseeable improvements in our approach, and explore future avenues of research in Chapter 5.

CHAPTER 2
Linear programming approaches to the SDP based on a semi-infinite formulation

2.1 Abstract

Interior point methods, the traditional methods for the SDP, are fairly limited in the size of problems they can handle. This chapter deals with an LP approach to overcome some of these shortcomings. We begin with a semi-infinite linear programming formulation of the SDP and discuss the issue of its discretization in some detail. We further show that a lemma due to Pataki on the geometry of the SDP, implies that no more than $O(\sqrt{k})$ (where k is the number of constraints in the SDP) linear constraints are required. To generate these constraints we employ the spectral bundle approach due to Helmberg and Rendl. This scheme recasts any SDP with a bounded primal feasible set as an eigenvalue optimization problem. These are convex non-smooth problems that can be tackled by bundle methods for non-differentiable optimization. Finally we present the rationale for using the columns of the bundle P maintained by the spectral bundle approach, as our linear constraints. We present numerical experiments that demonstrate the efficiency of the LP approach on two combinatorial examples, namely the max cut and min bisection problems.

The LP approach potentially allows one to approximately solve large scale semidefinite programs using state of the art linear solvers. Moreover one can incorporate these linear programs in a branch and cut approach for solving large scale integer programs.

2.2 The semidefinite programming problem

Consider the semidefinite programming problem

$$\begin{array}{lrcll} \min & C \bullet X & & & \\ \text{subject to} & \mathcal{A}(X) & = & b & (SDP) \\ & X & \succeq & 0, & \end{array}$$

with dual

$$\begin{array}{lllllll} \max & & b^T y & & & & \\ \text{subject to} & \mathcal{A}^T y & + & S & = & C & \qquad (SDD) \\ & & & S & \succeq & 0 & \end{array}$$

where $X, S \in \mathcal{S}^n$. Note that the convex constraint $X \succeq 0$ is equivalent to

$$d^T X d \;=\; dd^T \bullet X \;\geq\; 0 \quad \forall d \in B \tag{2.1}$$

where B is the compact set $\{d : ||d|| \leq 1\}$. These constraints are linear inequalities in the matrix variable X, but there is an *infinite* number of them. Thus SDP is a *semi-infinite* linear programming problem in $I\!R^{\frac{n(n+1)}{2}}$. The term semi-infinite programming derives from the fact that the LP has finitely many variables, with an infinite number of constraints. The survey paper by Hettich and Kortanek [55] discusses theory, algorithms, and applications of semi-infinite programming. One can also deduce that the primal SDP cone $\{X : X \succeq 0\}$ is *convex* from this semi-infinite formulation.

Since $d^T X d \geq 0$ can be rewritten as $\mathrm{tr}(dd^T X) \geq 0$, the definition of positive semidefiniteness immediately gives the following :

Corollary 2 *The symmetric $n \times n$ matrix S is positive semidefinite if and only if $S \bullet M \geq 0$ for all symmetric rank one matrices M.*

This chapter is organized as follows : section 2.3 discusses duality theory in SDP, in particular the notion of strong duality, and conditions under which this holds, section 2.4 deals with a semi-infinite LP formulation of the SDP, and the issue of its discretization. Section 2.5 deals with the geometry of the SDP; in particular we discuss the notions of basic feasible solutions, complementarity, non-degeneracy, and finally a lemma due to Pataki on the rank of optimal matrices in SDP. We present the spectral bundle approach in section 2.6, a fast algorithmic procedure for generating the desired constraints. Section 2.7 presents the rationale for using the columns of the bundle P, as linear constraints and introduces the max-cut and min-bisection problems, two combinatorial problems on which we test the LP approach. Section 2.8 presents computational results, and we conclude with some observations

and acknowledgments in sections 2.9 and 2.10 respectively. Most of the material in this chapter is borrowed from two papers (Krishnan and Mitchell [69, 70]). A glossary of all the notation in this chapter can be found in Appendix A.

2.3 Duality theory in SDP

We discuss the duality theory in semidefinite programming, i.e. the relations between (SDP) and (SDD). In the previous chapter, we considered weak duality, and in this section we give conditions under which strong duality holds. As we shall see, strong duality is intimately related to discretization of the semi-infinite LP formulation of the SDP we consider in the next section. Most of the material in this section can be found in the survey paper by Todd [110]. A general notion of strong duality in conic programming, of which the SDP is a special case, appears in Renegar [104].

There are a number of things that can go wrong in the context of duality in SDP. Two things can happen : non-attainment in either one of the primal and dual problems, and attainment in both sides but with a finite duality gap. We first present two examples to demonstrate these pathological cases. This is unlike the LP case where we always have attainment in both sides and strong duality, if either the primal or the dual has a finite objective value.

We illustrate this with a couple of examples. Consider

$$
\begin{array}{lrcl}
\min & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bullet X & & \\
\text{s.t.} & \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \bullet X & = & 0 \\
& \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix} \bullet X & = & 2 \\
& X & \succeq & 0
\end{array}
$$

with dual

$$\begin{array}{lll} \max & 2y_2 & \\ \text{s.t.} & S & = \begin{pmatrix} -y_1 & -y_2 & 0 \\ -y_2 & 0 & 0 \\ 0 & 0 & 1-2y_2 \end{pmatrix} \\ & S & \succeq \quad 0 \end{array}$$

From the primal constraints we have $X_{11} = 0$, and $X_{12} + X_{33} = 1$. Thus any feasible X is of the form $\begin{pmatrix} 0 & \xi_1 & \xi_2 \\ \xi_1 & \xi_3 & \xi_4 \\ \xi_2 & \xi_4 & 1-\xi_1 \end{pmatrix}$ (we exploit the fact that X is a symmetric matrix here). Also X must be positive semidefinite, and this forces ξ_1, $\xi_2 = 0$ (since $\begin{vmatrix} 0 & \xi_1 \\ \xi_1 & \xi_3 \end{vmatrix} \geq 0$, and $\begin{vmatrix} 0 & \xi_2 \\ \xi_2 & 1-\xi_1 \end{vmatrix} \geq 0$).

Thus any feasible $X = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \xi_3 & \xi_4 \\ 0 & \xi_4 & 1 \end{pmatrix}$. Thus the optimal solution of the primal is 1, and one such optimal solution is $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

In the dual we require $S = \begin{pmatrix} -y_1 & -y_2 & 0 \\ -y_2 & 0 & 0 \\ 0 & 0 & 1-2y_2 \end{pmatrix} \succeq 0$. So $y_2 = 0$ (since $\begin{vmatrix} -y_1 & -y_2 \\ -y_2 & 0 \end{vmatrix} \geq 0$). Moreover y_1 has to be nonpositive. Thus $y = (0; 0)$ is optimal, and the dual optimal value is 0. Note that both problems attain their optimal solutions, but there is a duality gap.

Also, we can construct examples such that there is an infinite duality gap between the primal and dual objective values. We wish to illustrate another phenomenon which could occur, which is again unlike the LP case.

Consider

$$\begin{array}{lrcl} \min & \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \bullet X & & \\ \text{s.t.} & \begin{pmatrix} -1 & 0 \\ 0 & 0 \end{pmatrix} \bullet X & = & -1, \\ & \begin{pmatrix} 0 & 0 \\ 0 & -1 \end{pmatrix} \bullet X & = & 0, \\ & X & \succeq & 0 \end{array}$$

with dual

$$\begin{array}{lrcc} \max & -y_1 & & \\ \text{s.t.} & S & = & \begin{pmatrix} y_1 & 1 \\ 1 & y_2 \end{pmatrix} \\ & S & \succeq & 0 \end{array}$$

It can be easily seen that the primal has only one feasible point $X = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$. Hence this is also the optimal solution, with an objective value of 0. Now consider the dual. We require $\begin{pmatrix} y_1 & 1 \\ 1 & y_2 \end{pmatrix} \succeq 0$. So the feasible region is $\{(y_1, y_2) : y_1 > 0, y_2 > 0, y_1 y_2 \geq 1\}$. So the optimal value is 0, but it is not attained. We can get arbitrarily close with solutions $(\epsilon, \frac{1}{\epsilon})$ for some arbitrarily small $\epsilon > 0$. Here there is no duality gap, but one of the optimal values is not attained.

We now turn to conditions which ensure strong duality, and existence of primal and dual solutions. Consider the following regions

$$\begin{array}{rcc} F(P) & = & \{X \in \mathcal{S}^n : \mathcal{A}(X) = b, X \succeq 0\} \\ F^0(P) & = & \{X \in F(P) : X \succ 0\} \\ F(D) & = & \{(y, S) \in \mathcal{R}^k \times \mathcal{S}^n : \mathcal{A}^T y + S = C : S \succeq 0\} \\ F^0(D) & = & \{(y, S) \in F(D) : S \succ 0\} \end{array} \tag{2.2}$$

Here $F(P)$, and $F(D)$ are the feasible regions for (SDP) and (SDD) respectively. Also, $F^0(P)$ and $F^0(D)$ are the strict interior of these feasible regions. Note that $F^0(P)$ and $F^0(D)$ are open sets (they have no boundary).

Theorem 8 *Suppose $F(P)$ and $F^0(D)$ are nonempty. Then (SDP) has a nonempty compact set of optimal solutions, and the optimal values of (SDP) and (SDD) are equal.*

The proof involves concepts from convex analysis, and can be found in Todd [110].

We can then obtain the following corollaries.

Corollary 3 *Suppose $F^0(P)$ and $F(D)$ are nonempty. Then (SDD) has a nonempty compact set of optimal solutions, and there is no duality gap*

Corollary 4 *Suppose both $F^0(P)$ and $F^0(D)$ are nonempty. Then each has a nonempty compact set of optimal solutions, and there is no duality gap.*

This is similar to the Slater constraint qualification in nonlinear programming. Also if the primal has a Slater point, then the dual attains its optimal solution; there is no duality gap and vice versa. Returning to our first example we find that $X = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \xi_3 & \xi_4 \\ 0 & \xi_4 & 1 \end{pmatrix}$, and $S = \begin{pmatrix} -y_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ force these matrices to be singular, and thus this problem has no strictly feasible primal or dual Slater points, which explains the duality gap. Also, in the second example, the primal has no Slater point since the only feasible point is $X = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, which is clearly singular. Thus from Corollary 3 we find that the dual may not attain its optimal solution. On the other hand, the dual has a Slater point, and hence the primal has a nonempty set of optimal solutions namely the singleton $X = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$. Henceforth we shall make the following assumption 1 with regard to the SDP's we handle.

Assumption 1 *Both (SDP) and (SDD) have strictly feasible points, namely the sets $\{X \in \mathcal{S}^n : \mathcal{A}(X) = b, X \succ 0\}$ and $\{(y, S) \in I\!R^k \times \mathcal{S}^n : \mathcal{A}^T y + S = C, S \succ 0\}$ are nonempty.*

Corollary 4 then guarantees that both (SDP) and (SDD) attain their optimal solutions X^* and (y^*, S^*), and that the duality gap $X^*S^* = 0$ at optimality, i.e. $C \bullet X^* = b^T y*$ (the objective values are the same).

We will also make the following assumption.

Assumption 2

$$\mathcal{A}(X) \;=\; b \quad \textit{implies} \quad trX \;=\; a \tag{2.3}$$

for some constant $a \geq 0$.

This assumption will be relevant to the discussion on the spectral bundle method (section 2.6). Helmberg [45] shows that every SDP whose primal feasible set is bounded can be rewritten to satisfy this assumption. We will also make use of this assumption, when we consider the issue of updating lower bounds for our cutting plane approach (see section 3.5.1).

2.4 A semi-infinite LP formulation of the SDP

The only nonlinearity in (SDP) and (SDD) is the requirement that the matrices X and S need to be positive semidefinite. As we have seen in the previous section these convex constraints are equivalent to an infinite number of linear constraints, giving rise to semi-infinite linear programs. We now consider two semi-infinite linear programs $(PSIP)$ and $(DSIP)$ for (SDP) and (SDD) respectively. These formulations follow directly from Corollary 2.

$$\begin{array}{lllll} \min & C \bullet X & & & \\ \text{subject to} & \mathcal{A}(X) & = & b & \qquad (PSIP) \\ & d^T X d & \geq & 0 \quad \forall d \in B & \end{array}$$

$$\begin{array}{lllll} \max & b^T y & & & \\ \text{subject to} & \mathcal{A}^T y + S & = & C & \qquad (DSIP) \\ & d^T S d & \geq & 0 \quad \forall d \in B & \end{array}$$

Here B is a compact set, typically $\{d : ||d||_2 \leq 1\}$ or $\{d : ||d||_{\text{inf}} \leq 1\}$. These are commonly used in *trust region methods*. (Conn et al [24]). A few remarks are now in order.

1. Since X is $n \times n$ and symmetric, $(PSIP)$ is a semi-infinite linear program in $\binom{n+1}{2} = \frac{n(n+1)}{2} = O(n^2)$ variables.

2. There are k variables in the semi-infinite formulation $(DSIP)$. We have $k \leq \binom{n+1}{2}$ (since the matrices A_i, $i = 1, \ldots, k$ are linearly independent).

3. It is more efficient to deal with the dual semi-infinite formulation, since we are dealing with smaller LP's.

We shall henceforth refer to $(DSIP)$ as (LDD).

We discuss the finite linear programs (LDR) and (LPR) and some of their properties below. Given a finite set of vectors $\{d_i, i = 1, \ldots, m\}$, we obtain the relaxation

$$\begin{array}{lll} \max & b^T y & \\ \text{subject to} & d_i d_i^T \bullet \mathcal{A}^T y \ \leq \ d_i d_i^T \bullet C & \text{for } i = 1, \ldots, m. \qquad (LDR) \end{array}$$

We now derive the linear programming dual to (LDR). We have

$$\begin{aligned} d_i d_i^T \bullet \mathcal{A}^T y &= d_i d_i^T \bullet (\sum_{j=1}^{k} y_j A_j) \\ &= \sum_{j=1}^{k} y_j d_i^T A_j d_i. \end{aligned}$$

Thus, the constraints of (LDR) can be written as

$$\sum_{j=1}^{k} y_j d_i^T A_j d_i \leq d_i^T C d_i \quad \text{for } i = 1, \ldots, m.$$

It follows that the dual problem is

$$\begin{array}{lrcl} \min & \sum_{i=1}^{m} d_i^T C d_i x_i & & \\ \text{subject to} & \sum_{i=1}^{m} d_i^T A_j d_i x_i & = & b_j \quad \text{for } j = 1, \ldots, k \\ & x & \geq & 0. \end{array}$$

This can be rewritten as

$$\begin{array}{lrcll} \min & C \bullet (\sum_{i=1}^m x_i d_i d_i^T) & & & \\ \text{subject to} & \mathcal{A}(\sum_{i=1}^m x_i d_i d_i^T) & = & b & (LPR) \\ & x & \geq & 0. & \end{array}$$

Theorem 9 *Any feasible solution x to (LPR) will give a feasible solution X to (SDP).*

Proof: This lemma follows directly from the fact that (LPR) is a constrained version of (SDP). However we present a formal proof. Set $X = \sum_{i=1}^m x_i d_i d_i^T$. From (LPR) it is clear that this X satisfies $\mathcal{A}X = b$. Moreover X is psd. To see this

$$\begin{array}{rcccl} d^T X d & = & d^T(\sum_{i=1}^m x_i d_i d_i^T) d & = & \sum_{i=1}^m x_i (d_i^T d)^2 \\ & \geq & 0 & \forall d & \end{array}$$

where the last inequality follows from the fact that $x \geq 0$. □

Corollary 5 *If the d_i's in $X = \sum_{i=1}^m x_i d_i d_i^T$ are normalized to have unit norm, then $\sum_{i=1}^m x_i = a$.*

We now discuss the issue of discretization of (LDD) in some detail. Consider the dual (LDP) to (LDD). It has the following form :

$$\begin{array}{lrcll} \min & \int_B x(\omega)(d(\omega)^T C d(\omega))\,\mathrm{d}\omega & & & \\ \text{subject to} & \int_B x(\omega)(d(\omega)^T A_i d(\omega))\,\mathrm{d}\omega & = & b_i \quad i = 1, \ldots, k & (LDP) \\ & x(\omega) & \geq & 0 & \end{array}$$

We have permitted an infinite number of vectors $z \in I\!R^k$ whose ith component is given by $d^T A_i d$, $i = 1, \ldots, k$ in the representation of $b \in I\!R^k$. However the *reduction* theorem stated below, indicates that at most k vectors z_i are required in the representation of b. For a constructive proof see Glashoff and Gustafson [35].

Theorem 10 *Let the vector $z \in \mathbb{R}^k$ be a nonnegative linear combination of the m vectors $z_1, \ldots, z_m \in \mathbb{R}^k$, i.e.*

$$z \;=\; \textstyle\sum_{i=1}^m x_i z_i, \quad x_j \geq 0, \quad j = 1, \ldots, m$$

Then z admits a representation

$$z \;=\; \textstyle\sum_{i=1}^m \bar{x}_i z_i, \quad \bar{x}_j \geq 0, \quad j = 1, \ldots, m$$

such that at most k of the numbers $\bar{x}_i$, $i = 1, \ldots, m$ are nonzero and such that the set of vectors $\{z_i | \bar{x}_i > 0\}$, is linearly independent.

In fact Caratheodory's theorem (see Urruty and Lemarechal [56]) is a special case of the reduction theorem. Here we apply the reduction theorem on $k+1$ equations, $z = \sum_{i=1}^m x_i z_i$, and $\sum_{i=1}^m x_i = 1$, and hence require $k+1$ points z_i, $i = 1, \ldots, k+1$ in the convex hull representation of z.

We cannot immediately conclude from theorem 10 that we only need to consider feasible solutions $\{x_1, \ldots, x_m\}$ with $m \leq k$, and that we can put $m = k$ from the start. It is quite possible that in the transition from (LDD) to (LDR) the value of the objective function is affected, i.e.

$$\textstyle\int_B x(\omega)(d(\omega)^T C d(\omega))\,\mathrm{d}\omega \;<\; \sum_{j=1}^k (d_j^T C d_j) x_j$$

Thus we should apply the reduction theorem on $k+1$ equations with the $(k+1)$th constraint being

$$\textstyle\int_B x(\omega)(d(\omega)^T C d(\omega))\,\mathrm{d}\omega \;=\; b_0$$

We obtain the important result that $m = k+1$ points d_i are *enough*. However if we know that (SDP) has a solution (Assumption 1), then we can put $m = k$ from the outset (Glashoff and Gustafson [35]). Thus an optimal solution to (LDP) has a finite support i.e. there are only a finite number of components m of x that are nonzero.

We give conditions ensuring that there exists a discretization of (LDD) with the same optimal value.

Theorem 11 *Suppose that the optimal value of the linear semi-infinite programming problem* (LDD) *is finite. If objective values of* (LDD) *and* (LDP) *are the same, i.e. there is no duality gap, and the dual problem* (LDP) *has an optimum solution, then* (LDD) *has a finite discretization* (LDR) *with the same optimal value.*

Proof: We have shown that an optimal solution to (LDP) has a finite support. Now let (LPR) and (LDR) be the discretizations of (LDP) and (LDD), respectively, corresponding to an optimal solution of (LDP) with a finite support. Thus we have $\text{val}(LDP) = \text{val}(LPR) = \text{val}(LDR)$, the latter equality following from the strong duality theorem in finite linear programming. Since the feasible set of (LDR) includes the feasible set of (LDD), $\text{val}(LDD) \geq \text{val}(LDR)$. Thus

$$\text{val}(LDD) \quad \geq \quad \text{val}(LDR) \quad = \quad \text{val}(LPR) \quad = \quad \text{val}(LDP)$$

which together with the assumption $\text{val}(LDD) = \text{val}(LDP)$ imply that $\text{val}(LDD) = \text{val}(LDR)$. □

In the presence of a duality gap we can show that

$$\text{val}(LDD) \quad > \quad \text{val}(LDR) \quad = \quad \text{val}(LPR) \quad = \quad \text{val}(LDP)$$

In this case $\text{val}(LDR)$ is a *sub-value* for (LDD) (see Anderson and Nash [6]). Moreover the duality gap can be as large as possible. See Anderson and Nash [6], Glashoff and Gustafson [35] and Bonnans and Shapiro [16] for examples of duality gaps in linear semi-infinite programming.

We now establish a bound on the number of constraints m in (LDR).

Theorem 12 *Suppose* (LDD) *is consistent and that there exists a finite discretization of* (LDD) *with the same optimal value. Then there exists a discretization* (LDR) *such that val*$(LDD) = val(LDR)$ *and* $m \leq k$.

Proof: Follows from Theorem 10. □

See Bonnans and Shapiro [16] for a proof based on Helly's theorem. Moreover if we were to solve (LDR) using a simplex scheme, then not more than k constraints

would be binding at optimality. The following results are only special cases for general results on the discretization of convex semi-infinite programs (Borwein [18], Ben Tal et al [13], Bonnans and Shapiro [16] and Hettich and Kortanek [55]).

2.5 Geometry of the semidefinite programming problem

In this case we discuss the geometry of the semidefinite programming problem. We first define complementarity and non-degeneracy in the context of semidefinite programming as a semi-infinite LP (see Anderson and Nash [6], also Alizadeh et al [4]). These non-degeneracy assumptions will imply a range of possible ranks for the primal and dual solutions X and S respectively. In particular, we will recover a theorem due to Pataki [99] on the rank of the primal matrix X. This has additional consequences on the issue of discretization of the semi-infinite LP presented in the previous section. Since Pataki's theorem plays an important role in this chapter, we shall present a short proof borrowing concepts from semi-infinite programming (also see Alizadeh et al [4], pages $9-10$). An alternate view of the geometry of semidefinite programming, using tools from convex analysis, can be found in Pataki [100].

The optimality conditions for the SDP include primal feasibility, dual feasibility and complementarity $XS = 0$. The complementarity condition implies that X and S commute, and so they share a common share of eigenvectors (see *simultaneous diagonalization* in Horn and Johnson [58]).

Thus, we have (Alizadeh et al [4]) :

Theorem 13 *Let X and (y, S) be primal and dual feasible respectively. Then they are optimal if and only if there exists $Q \in \mathcal{R}^{n\times n}$, with $Q^TQ = I$, and $\lambda, \omega \in \mathcal{R}_+{}^n$, such that*

$$X = QDiag(\lambda_1, \ldots, \lambda_n)Q^T \tag{2.4}$$

$$S = QDiag(\omega_1, \ldots, \omega_n)Q^T \tag{2.5}$$

$$\lambda_i \omega_i \quad = \quad 0, \quad i = 1, \ldots, n \tag{2.6}$$

all hold.

Here λ_i, ω_i, $i = 1, \ldots, n$ are the eigenvalues of X and S in the spectral decompositions (2.4) and (2.5) respectively. Also Q is an orthogonal matrix that contains the common set of eigenvectors corresponding to these eigenvalues. Thus if X has rank r, and S has rank s, then (2.6) implies $r + s \leq n$ (since λ_i and ω_i could be zero simultaneously).

We say that *strict complementarity* holds if $r + s = n$. Alizadeh et al [4] show that strict complementarity is a generic property of SDP's, i.e. occurs almost everywhere, except possibly on a set of measure zero.

We will now introduce some terminology which will enable us to define nondegeneracy, and the notion of basic feasible solutions in a general conic setting (these were introduced by Anderson and Nash [6] in a semi-infinite LP setting; since conic optimization problems can be written as semi-infinite linear programs these can be easily extended to such a setting). Consider the conic programming setting we introduced in section 1.2. The feasible region X of the primal conic program (CPP) is

$$F(CPP) \quad = \quad \{x \in \mathcal{R}^n : Ax = b, x \succeq_K 0\}$$

Note that we assume that $F(CPP) \subseteq \mathcal{R}^n$ for ease of exposition. We will assume that $A \in \mathcal{R}^{k \times n}$ has full row rank k.

Let

$$P \quad = \quad \{x : x \succeq_K 0\} \tag{2.7}$$

The letter P indicates that we are dealing with the primal cone.

Let us define two linear subspaces of $\mathcal{R}^n$ below :

$$\mathcal{N} \quad = \quad \{y \in \mathcal{R}^n : Ay = 0\} \tag{2.8}$$

$$\mathcal{B}_x \;=\; \{\psi \in \mathcal{R}^n : x + \lambda\psi, x - \lambda\psi \in P, \;\; \text{for some} \;\; \lambda > 0\} \tag{2.9}$$

To illustrate these concepts, let us consider the LP case first. In this case we are dealing with a set $F(LP) \subseteq \mathcal{R}^n$.

$$\begin{aligned} P^{lp} &= \{x \in \mathcal{R}^n : x \geq 0\} \\ \mathcal{B}^{lp}{}_x &= \{\psi \in \mathcal{R}^n : \psi_i = 0 \;\text{ if }\; x_i = 0, i = 1, \ldots, n\} \\ \mathcal{N}^{lp} &= \{y \in \mathcal{R}^n : Ay = 0\} \end{aligned}$$

Now if we pick an $x \in \mathcal{R}_+{}^n$ with its first r components strictly positive, and the rest zero, then $\dim(\mathcal{B}^{lp}{}_x) = r$, and $\dim(\mathcal{N}^{lp}) = n - k$ (since we are assuming that the rows of A are linearly independent).

Now consider the SDP case. Consider a matrix X of rank r, with spectral decomposition

$$X \;=\; Q\text{Diag}(\lambda_1, \ldots, \lambda_r)Q^T$$

with $Q^TQ = I_r$. Here we are dealing with a convex set $F(SDP) \subset \mathcal{S}^n$.

$$\begin{aligned} P_r^{sdp} &= \{X \in \mathcal{S}^n : X \succeq 0, \text{rank}(X) = r\} \\ \mathcal{B}^{sdp}{}_x &= \{Q \begin{bmatrix} U & 0 \\ 0 & 0 \end{bmatrix} Q^T : U \in \mathcal{S}^r\} \\ \mathcal{N}^{sdp} &= \{Y \in \mathcal{S}^n : A_i \bullet Y = 0, i = 1, \ldots, k\} \end{aligned} \tag{2.10}$$

The subscript r in P_r^{sdp} denotes that we are dealing with all psd X of rank r. To see the second relation, note that for ΔX in $\mathcal{B}^{sdp}{}_x$ we have

$$Q^T(X + \epsilon\Delta X)Q \;=\; \begin{bmatrix} \text{Diag}(\lambda_1, \ldots, \lambda_r) + \epsilon U & 0 \\ 0 & 0 \end{bmatrix}$$

so that $X \pm \epsilon\Delta X \in P_r$, for sufficiently small ϵ. Since any matrix in $\mathcal{B}^{sdp}{}_x$ is determined by the $r^{\bar{2}}$ parameters of U, $\dim(\mathcal{B}_X) = r^{\bar{2}}$. Also $\dim(\mathcal{N}^{sdp}) = n^{\bar{2}} - k$ (again we are assuming that the matrices $A_i, i = 1, \ldots, k$ are linearly independent in $\mathcal{S}^n$).

The Anderson and Nash [6] (pages $19-21$) non-degeneracy condition for a semi-infinite LP is

$$\mathcal{B}_x + \mathcal{N} \;=\; \mathcal{R}^n \tag{2.11}$$

Also, Anderson and Nash [6] define any feasible x to be basic if

$$\mathcal{B}_x \cap \mathcal{N} \;=\; \{0\} \tag{2.12}$$

The definition (2.12) is to ensure that every basic solution x is also an extreme point of the feasible region. In (CPP) we are minimizing a linear function over a convex set, and as in the LP case we expect our optimal solution to be one of the extreme points of this convex region. To see this, note that $x \in \mathcal{R}^n$ is not an extreme point of the primal feasible region $F(CPP)$ if and only if there exists a $\lambda > 0$, and $\psi \neq 0 \in \mathcal{R}^n$, such that $x + \lambda\psi$, and $x - \lambda\psi$ are both feasible. This is true if and only if

$$\begin{aligned} x + \lambda\psi &\in P \\ x - \lambda\psi &\in P \end{aligned}$$

and

$$\begin{aligned} A(x + \lambda\psi) &= A(x - \lambda\psi) \\ &= \quad Ax \quad = b \end{aligned}$$

Thus $\psi \in \mathcal{B}_x \cap \mathcal{N}$, implying that x is not basic. Now let us see what these conditions imply for the LP and the SDP. Consider the LP case first.

The nondegeneracy condition (2.11), for $x \in {\mathcal{R}_+}^n$, with $x_i > 0$, $i = 1, \ldots, r$, and $x_i = 0$, $i = r+1, \ldots, n$ gives

$$\dim({\mathcal{B}^{lp}}_x) + \dim(\mathcal{N}^{lp}) \leq n \quad \Rightarrow \quad r \leq k$$

On the other hand the condition (2.12) implies that $r \geq k$, so that both these conditions together imply that

$${\mathcal{B}^{lp}}_x \oplus \mathcal{N}^{lp} = \mathcal{R}^n \quad \Rightarrow \quad r = k$$

Thus we recover the fact that in a nondegenerate LP at a basic feasible solution x, exactly k components of x are nonzero.

Now let us consider the SDP case. Proceeding in the similar fashion for a psd X of rank r, we find (2.11) and (2.12) together imply that

$$\mathcal{B}^{sdp}{}_x \oplus \mathcal{N}^{sdp} = \mathcal{S}^n \quad \Rightarrow \quad r = \sqrt[2]{k}$$

Note that we choose r to be floor of the quantity on the right, if this quantity is not an integer. Alizadeh et al [4] introduce a different definition for primal and dual nondegeneracy. More details on this approach for nondegeneracy can be found in Bonnans and Shapiro [16]. We present their definitions, and show that Pataki's theorem [99] appears as a consequence of these definitions.

Definition 4 *Let P_r be cone of psd matrices X of rank r. Also, for a given $X \in P_r$, let $\mathcal{T}_X$ be the tangent space to P at X, i.e.*

$$\mathcal{T}_X \;=\; \{Q \begin{bmatrix} U & V \\ V^T & 0 \end{bmatrix} Q^T : U \in \mathcal{S}^r, V \in \mathcal{R}^{r\times(n-r)}\} \tag{2.13}$$

where Q is the orthogonal matrix of the eigenvectors of X. Also, let $\mathcal{N}$ be defined as in (2.8). Then a solution X to (SDP) is primal nondegenerate if

$$\mathcal{T}_X + \mathcal{N} \;=\; \mathcal{S}^n \tag{2.14}$$

One way to motivate this definition is via a perturbation argument : The set $\mathcal{B}_X$ is the cone of feasible directions to P_r at X, i.e. given an $X \in P_r$, the set $\mathcal{B}_X$ contains the set of feasible directions $\psi \in \mathcal{S}^n$, such that $X \pm \epsilon\psi$ is still contained in P_r. On the other hand the set $\mathcal{T}_X$ is the tangent plane to P_r at X, i.e. the set $\mathcal{T}_X$ contains the set of directions $\psi \in \mathcal{S}^n$, such that $\mathrm{dist}(X \pm \epsilon\psi, P_r) = O(\epsilon^2)$. To see this, consider sufficiently perturbations $X \pm \epsilon\Delta X$, with $\Delta X \in \mathcal{T}_X$. We have to guarantee that the resulting matrix is psd, and hence have to add the following matrix $W = Q \begin{pmatrix} 0 & 0 \\ 0 & D \end{pmatrix} Q^T$ to $X \pm \epsilon\Delta X$. Here $D \in \mathcal{S}^{n-r}$. The question then is how large can D get?. We utilize the Schur complement idea (Theorem 1) to obtain

a bound on D.

$$\begin{aligned} X + \epsilon\Delta X + W \quad &= \quad Q \begin{bmatrix} \text{Diag}(\lambda_1, \ldots, \lambda_r) + \epsilon U & \epsilon V \\ \epsilon V^T & D \end{bmatrix} Q^T \\ X + \epsilon\Delta X + W \succeq 0 \quad &\Leftrightarrow \quad \begin{bmatrix} \text{Diag}(\lambda_1, \ldots, \lambda_r) + \epsilon U & \epsilon V \\ \epsilon V^T & D \end{bmatrix} \succeq 0 \\ &\Leftrightarrow \quad D - \epsilon^2 V^T (\text{Diag}(\lambda_1, \ldots, \lambda_r) + \epsilon U)^{-1} V \succeq 0 \end{aligned}$$

We have utilized the fact that $\text{Diag}(\lambda_1, \ldots, \lambda_r)$ (since it is positive definite) dominates ϵU. Thus this matrix $\text{Diag}(\lambda_1, \ldots, \lambda_r) + \epsilon U$ is invertible in the Schur complement. Thus loosely speaking $||D|| = O(\epsilon^2)$. For a sufficiently small perturbation ϵ we can say that whenever $\Delta X \in \mathcal{T}_X$, then the matrix $X \pm \epsilon\Delta X$ is sufficiently close to being a psd matrix of rank r.

Similarly dual nondegeneracy is defined as follows :

Definition 5 *Let D_s be the cone of positive semidefinite matrices S of rank s. Consider the following spectral decomposition of S*

$$S \quad = \quad QDiag(0, \ldots, 0, w_{n-s+1}, \ldots, w_n)Q^T$$

Also, define tangent space to D_s at S as

$$\mathcal{T}_S \quad = \quad \{Q \begin{bmatrix} 0 & V \\ V^T & W \end{bmatrix} Q^T : V \in \mathcal{R}^{(n-s)\times s}, W \in \mathcal{S}^s\} \tag{2.15}$$

The point (y, S) is dual nondegenerate if it is dual feasible and S satisfies

$$\mathcal{T}_S + span(A_k) \quad = \quad \mathcal{S}^n \tag{2.16}$$

Theorem 14 *Suppose, X and (y, S) are respectively, primal and dual nondegenerate, and optimal, with rank$(X) = r$, and rank$(S) = s$. Then*

$$n - \sqrt[5]{n^5 - k} \quad \leq \quad r \quad \leq \quad \sqrt[5]{k} \tag{2.17}$$

and

$$n - \sqrt[\bar{2}]{k} \;\leq\; s \;\leq\; \sqrt[\bar{2}]{n^{\bar{2}} - k} \tag{2.18}$$

Proof: Consider the primal nondegeneracy condition (2.14) first. Since $\dim(\mathcal{T}_X) = n^{\bar{2}} - (n-r)^{\bar{2}}$, and $\dim(\mathcal{N}) = n^{\bar{2}} - k$, we have $(n-r)^{\bar{2}} \leq n^{\bar{2}} - k$, and hence the lower bound in (2.17). Similarly using $\dim(\mathcal{T}_Z) = n^{\bar{2}} - (n-s)^{\bar{2}}$ (after some simplifications), and the dual nondegeneracy condition (2.16) gives $(n-s)^{\bar{2}} \leq k$. This gives the lower bound in (2.18). Using the complementarity condition $r + s \leq n$, then gives the upper bounds in (2.17) and (2.18) respectively. □

Primal (dual) nondegeneracy imply the uniqueness of the dual (primal) solutions. In fact we have (Alizadeh et al [4])

Theorem 15 *Let X be primal nondegenerate and attain its optimal solution. Then there exists a unique dual optimal solution (y, S). Also, if (y, S) is dual nondegenerate and optimal, then there exists a unique primal optimal solution.*

The converse of Theorem 15 holds under an additional assumption of strict complementarity (Alizadeh et al [4]). Again, this is unlike the LP case where both sides of Theorem 15 always hold, regardless of strict complementarity.

The upper bound on X in Theorem 14 will be important in the succeeding discussions. Hence we will restate it as theorem in itself.

Theorem 16 *There exists an optimal solution X^* with rank r satisfying the inequality $\frac{r(r+1)}{2} \leq k$. Here k is the number of constraints in (SDP).*

Theorem 16 suggests that there is an optimal matrix X that satisfies the upper bound (whose rank is around $O(\sqrt{k})$). It must be emphasized that Pataki [99, 100] derived Theorem 16 under no nondegeneracy assumptions. However, without this assumption the bound need not hold for all solutions.

Thus the optimal X can be expressed as $X = P\Lambda P^T$. Here $\Lambda \in \mathcal{S}^r$ is a diagonal matrix containing the r nonzero eigenvalues of X, and P is an orthonormal matrix satisfying $P^T P = I_r$, and containing the eigenvectors, corresponding to these r

eigenvalues. Also, noting that $r = \sqrt[2]{k}$, is understood to mean the positive square root, we have

$$r \leq \lfloor \frac{\sqrt{1+8k}-1}{2} \rfloor$$

The quantity on the right is an upper bound on the value of r we need, if we are to preserve the set of optimal solutions. The floor operation is required since this quantity need not be an integer. In the context of solving an (SDP) as an (LP), Pataki's theorem suggests that there is an relaxation (LDR) which exactly captures the (SDP) objective value, and has no more than $O(\sqrt{k})$ constraints. This is a tightening of Theorem 12, which relies solely on linear semi-infinite programming theory. As a result not more than $O(\sqrt{k})$ variables in (LPR) would be nonzero. We express this fact in corollary 6 below.

Corollary 6 *Any (LPR) with val(LPR) = val(SDP) must be degenerate.*

Theorem 17 *Let $X = P\Lambda P^T$, and let p_i, $i = 1, \ldots, r$ be the columns of P corresponding to the r nonzero eigenvalues in Λ. Then (SDP) is equivalent to (LDR) with $d_i = p_i$, $i = 1, \ldots, r$.*

Proof: The optimal value to (LPR) gives an upper bound on the optimal value of (SDP). Thus an optimal solution to (SDP) is also optimal in (LPR), provided it is feasible in (LPR). The optimal solution to (SDP) is given by $X = P\Lambda P^T = \sum_{i=1}^{r} \lambda_i p_i p_i^T$, where $\lambda_i > 0$, $i = 1, \ldots, r$, and p_i, $i = 1, \ldots, r$ are the corresponding eigenvectors. This is clearly feasible in (LPR). This corresponds to (LDR) with $d_i = p_i$, $i = 1, \ldots, r$. □

Theorem 17 tells us precisely what constraints we should looking for in our LP relaxations, namely those in the *null space of the optimal dual slack matrix S*.

We must mention at this stage that we could in practice solve (SDP) using an interior point scheme, and utilize the P corresponding to the strictly positive eigenvalues of X as d. This would give an (LDR) which attains the (SDP) optimal value.

2.6 The spectral bundle method

The spectral bundle method is due to Helmberg and Rendl [52]. Other references include Helmberg et al [45, 50, 51] and Oustry [93]. In this section we give a detailed description of the spectral bundle scheme.

Since our original (SDP) is a minimization problem, we will be dealing in this section, with the minimum eigenvalue function, a concave function. However we shall be using terms like subgradients, subdifferential etc, usually associated with convex functions. These terms should be understood to be the corresponding analogues for a concave function.

Consider the eigenvalue optimization problem (2.19).

$$\max_y \quad a\lambda_{min}(C - \mathcal{A}^T\mathrm{y}) \quad + \quad b^T\mathrm{y} \tag{2.19}$$

Problems of this form are equivalent to the dual of semidefinite programs (SDP), whose primal feasible set has a constant trace (Assumption 2). This can be easily verified as follows. From the variational characterization of the minimum eigenvalue function (see section 1.5.3), we have

$$\lambda_{min}(C - \mathcal{A}^T y) \quad = \quad \min_{X:\mathrm{tr}X=1, X\succeq 0}(C - \mathcal{A}^T y) \bullet X \tag{2.20}$$

Thus (2.19) is equivalent to taking the Lagrangian dual of (SDP) with y being the vector of dual variables corresponding to $\mathcal{A}(X) = b$, and observing that $a\lambda_{min}(C-\mathcal{A}^T\mathrm{y}) = \min_{X:\mathrm{tr}X=a, X\succeq 0}(C-\mathcal{A}^T y)\bullet X$. We can rewrite (2.19) strictly as an eigenvalue optimization problem by incorporating the linear term $b^T y$ into the eigenvalue function (each A_i is now replaced with $(A_i - b_i I)$). Hereafter $\lambda_{min}(C - \mathcal{A}^T y)$ is actually the entire objective function $\lambda_{min}(C - \mathcal{A}^T y) + b^T y$. It can be shown that any SDP with a bounded feasible set can be written in this way, i.e. as an eigenvalue optimization problem. The minimum eigenvalue function $\lambda_{min}(.)$ is a non-smooth concave function (the concavity follows from the variational characterization). In fact this function is non-smooth precisely at those points where the minimum eigenvalue has a multiplicity greater than one. To see this, consider a

general example with $(C - \mathcal{A}^T y) = \begin{bmatrix} 1+y_1 & y_2 \\ y_2 & 1-y_1 \end{bmatrix}$. It can be easily verified that $\lambda_{min}(C - \mathcal{A}^T y) = (1 - \sqrt{y_1^2 + y_2^2})$. This function is nonsmooth at $y = 0$ (see figure 2.1). Henceforth we denote $S = (C - \mathcal{A}^T y)$. A general scheme to minimize such

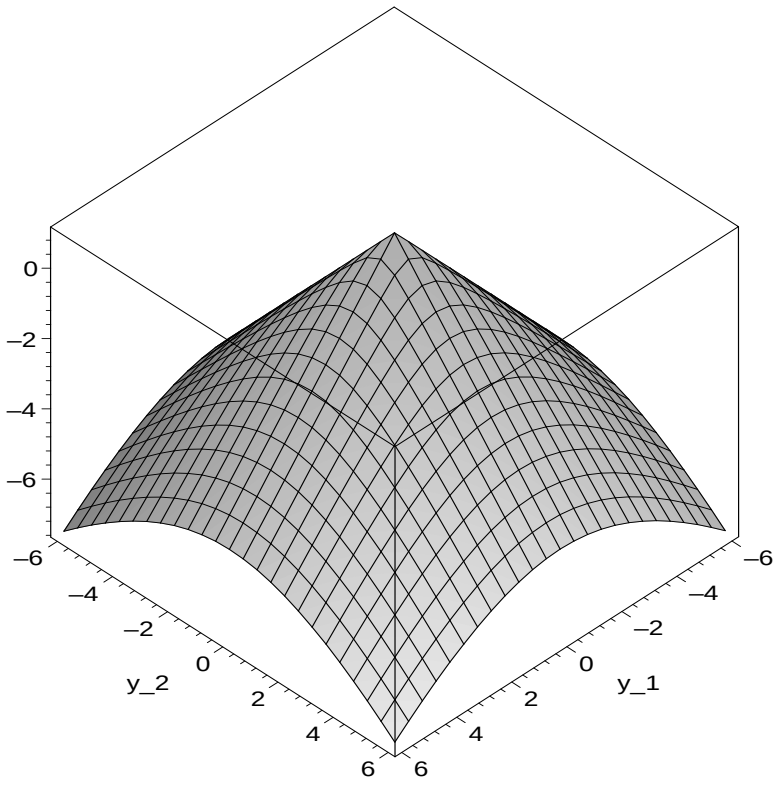


Figure 2.1: Nonsmooth nature of the minimum eigenvalue function

functions is the *bundle* scheme (Kiwiel [65, 66, 67], Lemarechal [75] and Schramm et al [107] and the books by Urruty and Lemarechal [56, 57] (especially Chapter $XV(3)$). An excellent survey on eigenvalue optimization appears in Lewis and Overton [76]. We will find the expression (2.21) for a concave function $f : \mathcal{R}^n \rightarrow \mathcal{R}$, and it subdifferential $\partial f(x)$ very useful in the succeeding discussions.

$$\begin{aligned} f(x) &= \min\{{\alpha_i}^T x + \beta_i : i \in \mathcal{I}\} \\ \partial f(x) &= \text{conv}\{\alpha_i : i \in \mathcal{I}, f(x) = {\alpha_i}^T x + \beta_i\} \end{aligned} \tag{2.21}$$

The expression for $f(x)$ follows from the characterization of a concave function, as the pointwise minimum of a set of linear functions. Also $\mathcal{I}$ is a discrete set, and

conv denotes the convex hull operation. See Rockafellar [105] for more details.

To motivate the bundle approach, we begin with some preliminaries for the minimum eigenvalue function

The minimum eigenvalue function $\lambda_{min}(S)$ has following representation, due to its variational characterization

$$\lambda_{min}(S) \quad = \quad \min_{||p||=1} pp^T \bullet S \tag{2.22}$$

The minimum in (2.22) is attained at any p, which is a normalized eigenvector corresponding to $\lambda_{min}(S)$. Thus the subdifferential for $\lambda_{min}(S)$ is given by

$$\partial\lambda_{min}(S) \quad = \quad \text{conv}\{pp^T : p \text{ a normalized eigenvector to } \lambda_{min}(S)\} \tag{2.23}$$

This follows directly from (2.21) and (2.22). Also note that $pp^T \bullet S = p^T Sp = \lambda_{min}(S)$, for p in (2.23). For our purposes, we will work with the following alternate expression for $\partial\lambda_{min}(S)$ given in Theorem 18. The elegant proof is due to Overton [95].

Theorem 18 *Suppose the minimum eigenvalue $\lambda_{min}(S)$ has multiplicity r, i.e. the eigenvalues of S are ordered as*

$$\lambda_1(S) = \ldots = \lambda_r(S) < \lambda_{r+1}(S) \leq \ldots \leq \lambda_n(S)$$

Then the subdifferential of $\lambda_{min}(S)$ at S is the set

$$\partial\lambda_{min}(S) \quad = \quad \{U = P_1 V {P_1}^T : V \in \mathcal{S}^r, tr(V) = 1, V \succeq 0\} \tag{2.24}$$

where the columns of P_1 contain a orthonormal basis for the eigenspace of $\lambda_{min}(S)$.

Proof: Consider the following spectral decomposition of S

$$S = P\Lambda P^T \quad P = [P_1, P_2]$$

Now from section 1.5.3, we can express $\lambda_{min}(S)$ as the following SDP

$$\lambda_{min}(S) = \min_{W:\mathrm{tr}(W)=1, W\succeq 0} W \bullet S \tag{2.25}$$

It is clear that the choice

$$\begin{aligned} W &= P \begin{bmatrix} V & 0 \\ 0 & 0 \end{bmatrix} P^T \\ &= P_1 V {P_1}^T \end{aligned}$$

attains the minimum in (2.25), since $\Lambda(1:r, 1:r) = \lambda_1(S) I_r$ contains the smallest components of Λ. The subdifferential can be computed from (2.21), and is :

$$\begin{aligned} \partial\lambda_{min}(S) &= \mathrm{conv}\{U = P_1 V {P_1}^T : V \in \mathcal{S}^r, \mathrm{tr}(V) = 1, V \succeq 0\} \\ &= \{P_1 V {P_1}^T : V \in \mathcal{S}^r, \mathrm{tr}(V) = 1, V \succeq 0\} \end{aligned}$$

No convex hull operation conv is required since the set is already convex. □

An upper bound on r is given by Pataki's theorem (Theorem 16). We expect the value of r to increase as we approach optimality. This is because we are maximizing the minimum eigenvalue and the eigenvalues tend to cluster together. Once we have the subdifferential we can derive the directional derivative $\lambda'_{min}(y, D)$ as follows :

$$\begin{aligned} \lambda'_{min}(y, D) &= \min_{W \in \partial\lambda_{min}(S)} W \bullet D \\ &= \min_{\mathrm{tr}(V)=1, V\succeq 0} (PVP^T) \bullet D \\ &= \min_{\mathrm{tr}(V)=1, V\succeq 0} V \bullet (P^T D P) \\ &= \lambda_{min}(P^T D P) \end{aligned} \tag{2.26}$$

Here $D = -\mathcal{A}^T d = \sum_{i=1}^k d_i(b_i I - A_i)$ where d is the current search direction. P is the orthogonal matrix containing a basis for the eigenspace corresponding to $\lambda_{min}(S)$. The first equality follows from the fact that the directional derivative is the support function for the subdifferential (Urruty and Lemarechal [56] and Oustry [93]). The second equality follow from Theorem 18. The third equality follows from $\mathrm{tr}(AB) = \mathrm{tr}(BA)$, and the last equality from the expression for $\lambda_{min}(.)$ in section 1.5.3.

We can design a *steepest descent* like scheme for maximizing the minimum eigenvalue function as follows :- The search direction d involves solving the following subproblem

$$\begin{aligned} & \max_{d\in I\!R^k,||d||\leq 1} \lambda_{min}(P^TDP) \\ = & \max_{d\in I\!R^k,||d||\leq 1} \min_{\mathrm{tr}(W)=1,W\succeq 0}(P^TDP)\bullet W \\ = & \min_{\mathrm{tr}(W)=1,W\succeq 0} \max_{d\in I\!R^k,||d||\leq 1}(b-\mathcal{A}(PWP^T))^Td \\ = & \min_{\mathrm{tr}(W)=1,W\succeq 0} ||b-\mathcal{A}(PWP^T)|| \\ = & \min_{X\in\partial\lambda_{min}(S)} ||b-\mathcal{A}(X)|| \end{aligned} \tag{2.27}$$

The constraint $||d|| \leq 1$ in (2.27) ensures that solution to the max-min problem on the left is bounded. The first equality uses the variational characterization of the minimum eigenvalue function. The second equality is obtained by interchanging the max and min using the notion of strong duality. The third inequality follows from the fact that the maximum is attained at $d = \frac{(b-\mathcal{A}(PWP^T))}{||b-\mathcal{A}(PWP^T)||}$. The fourth equality follows from (2.24). Thus we have a minimum norm problem, which has a unique solution since are minimizing a strictly convex function, over a closed compact convex set. The last problem in (2.27) can be replaced by the following equivalent problem. This gives a quadratic semidefinite programming problem ($QSDP$). For the convergence of the scheme, see Lemarechal [75].

$$\min_{X\in\partial\lambda_{min}(S)} \frac{1}{2}||b-\mathcal{A}(X)||^2 \qquad (QSDP)$$

A few points are now in order :

1. At optimality when the search direction $d = 0$, we get primal feasibility $\mathcal{A}(X) = b$.

2. Solving ($QSDP$) with $X = PVP^T \in \partial\lambda_{min}(S)$ amounts to relaxing $S \succeq 0$ to $P^TSP \succeq 0$.

3. This amounts to the solution of the following relaxed eigenvalue problem (2.28)

$$\max_y \quad a\lambda_{min}(P^T(C-\mathcal{A}^Ty)P) + b^Ty \tag{2.28}$$

The spectral bundle method returns the objective value $b^T y + a\lambda_{min}(C - \mathcal{A}^T y)$, which is a lower bound on the objective value of (SDD). To see this, note that the bundle method returns the value $b^T \bar{y}$ corresponding to a feasible $\bar{y}$ (such that $\bar{S} = (C - \mathcal{A}^T \bar{y})$ is psd). In fact, we have

$$\bar{y} \quad = \quad y + \lambda_{min}(C - \mathcal{A}^T y)\hat{y}$$

where $\hat{y}$ satisfies Theorem 20 (see Section 3.5.1). Thus the spectral bundle method returns a lower bound on the optimal SDP objective value.

4. For the convergence of the scheme, we actually need to consider the entire ϵ subdifferential at each iteration which consists of all eigenvectors corresponding to eigenvalues which are within an ϵ of the minimum eigenvalue (for a formal definiton of the ϵ subgradient see Urruty and Lemarechal [56, 57]). So we really have to solve the following problem

$$\min_{X \in \partial_\epsilon \lambda_{min}(S)} \quad \tfrac{1}{2}||b - \mathcal{A}(X)||^2 \tag{2.29}$$

We must also mention that ϵ here, should not be thought as a small parameter. In fact this quantity should be suitably chosen in each iteration (see Lemarechal [75]). Thus this problem is about as hard as solving the original SDP.

Computing the entire ϵ subdifferential at each iteration is prohibitively expensive, and this is where the bundle idea comes in handy (Helmberg et al [52]). Instead of computing the entire subdifferential, we consider an arbitrary subgradient from the current subdifferential and utilize the important subgradients from the previous iterations. Helmberg and Rendl [52] consider the following subset (2.30) of the subdifferential. For more details also see Appendix B.

$$\begin{array}{rcc} \hat{\mathcal{W}} & = & \{W : W = \alpha \bar{W} + PVP^T, \alpha + \mathrm{tr}(V) = 1, \alpha \geq 0, V \succeq 0\} \\ & \approx & \partial\lambda_{min}(S) \end{array} \tag{2.30}$$

Here $\bar{W}$ is known as the *aggregate matrix* and contains the less important subgradient

information, while P is known as the *bundle* and contains the important subgradient information.

The bundle approach handles the steps in (2.27) differently. Firstly, the constraint $||d|| \leq 1$ is lifted into the objective function by means of a Lagrangian weight parameter u. (the term $\frac{u}{2}d^T d$ penalizes us from going too far from the current point). We repeat the steps in the derivation of $(QSDP)$ below.

$$\begin{array}{rcc} & \max_{d\in I\!R^k, ||d||\leq 1} \lambda_{min}(P^T DP) & \\ \geq & \max_{d\in I\!R^k} \lambda_{min}(P^T DP) - \frac{u}{2}d^T d & \\ = & \max_{d\in I\!R^k} \min_{\mathrm{tr}(W)=1, W\succeq 0}(P^T DP) \bullet W - \frac{u}{2}d^T d & \\ = & \min_{\mathrm{tr}(W)=1, W\succeq 0} \max_{d\in I\!R^k}(b - \mathcal{A}(PWP^T))^T d - \frac{u}{2}d^T d & (2.31) \\ = & \min_{\mathrm{tr}(W)=1, W\succeq 0} \frac{1}{2u}||b - \mathcal{A}(PWP^T)||^2 & \\ = & \min_{X\in\partial\lambda_{min}(S)} \frac{1}{2u}||b - \mathcal{A}(X)||^2 & \\ \approx & \min_{X\in\hat{\mathcal{W}}} \frac{1}{2u}||b - \mathcal{A}(X)||^2 \qquad (QSDP_{sb}) & \end{array}$$

The first inequality follows from Lagrangian duality (note that $u > 0$). The term $-\frac{u}{2}d^T d$ ensures that we do not stray too far from our current iterate. The next two equalities are obtained in the same way as in (2.27). For the third equality, note that the quantity in the max problem is a strictly concave function, and attains its maximum when

$$\begin{array}{rcl} d & = & \frac{1}{u}(b - \mathcal{A}(PWP^T)) \\ y^{new} & = & y^{old} + \frac{1}{u}(b - \mathcal{A}(PWP^T)) \end{array} \qquad (2.32)$$

Substituting this value of d and simplifying gives the third equality. The fourth equality follows from (2.24), and the last inequality is obtained from (2.30). This gives a quadratic semidefinite programming problem $(QSDP_{sb})$. In the last inequality, we are approximating the function being minimized, by its majorant, i.e. instead of optimizing over all psd matrices W of trace one, we are confining ourselves to a smaller set. However, for practical purposes the set $\hat{\mathcal{W}}$ is generally a good approximation to $\partial\lambda_{min}(S)$ in every iteration. If not, we enrich $\hat{\mathcal{W}}$ with more subgradient information termed as a *null step* in the approach.

We are now ready to describe the key ideas in the spectral bundle approach.

1. Consider a subset $X = PVP^T$ of the feasible X matrices. We are exploiting Pataki's lemma [99] by considering only the subset of all optimal X matrices, which allows us to operate in a lower dimensional space.

2. Use this X to improve y. This involves solving the following direction finding subproblem

$$\min_{X \in \hat{\mathcal{W}}} \quad \frac{1}{2u}||b - \mathcal{A}(X)||^2 \qquad (QSDP_{sb})$$

We are now optimizing over all X belonging to the set $\hat{\mathcal{W}}$ instead of the entire subdifferential. $(QSDP_{sb})$ is a quadratic SDP in $\binom{r+1}{2} + 1$ variables. Moreover the aggregate matrix $\bar{W}$ gives us the flexibility of using fewer columns (than the r given by Pataki's lemma) in P. This is important, for we want to keep r small so that $(QSDP_{sb})$ can be solved quickly. Also, (2.32) shows how y is updated using information from X (the term PWP^T in (2.32) is the current approximation to X).

3. Use the value of y in turn to improve X. This is done by updating the bundle P in each iteration. P attempts to recreate $\partial\lambda_{min}(S)$ at each iteration. If P is a bad approximation to the subdifferential at a point, we remain at that point and update P with more subgradient information. This is termed as a *null step* in the spectral bundle scheme. The bundle updates are described in (2.33) and (2.34) respectively.

Before we conclude our discussion on the spectral bundle method, here is the basic algorithm. We shall introduce some notation : let $f(x^k) = \lambda_{min}(C - \mathcal{A}^T x^k)$. Also let $\hat{f^k}(x^k)$ denote the objective value of $(QSDP_{sb})$ for some x^k. We will denote an iterate in a serious step as x. In an iteration k, if we decide to perform a serious step then $x^k = y^k$, else x^k remains at its earlier value x^{k-1} (see update rules 5 and 6 below).

1. **Initialization :** An initial point $y^0 \in \mathcal{R}^k$, an eigenvector v^0 corresponding to $\lambda_{min}(C - \mathcal{A}^T y^0)$, a termination parameter $\epsilon > 0$, a weight parameter $u > 0$, an improvement parameter $m_L \in (0, \frac{1}{2})$, and finally an upper bound on the number of columns r of P.
   Set $x^0 = y^0$, $P^0 = v^0$, $\bar{W}^0 = v^0(v^0)^T$

2. **Update y :** Solve $(QSDP_{sb})$ for α^k and V^k ($W^{k+1} = \alpha^k \bar{W}^k + P^k V^k P^{k^T}$). Compute y^{k+1} from (2.32). Decompose $V^k = Q_1 \Lambda_1 Q_1^T + Q_2 \Lambda_2 Q_2^T$, with $\mathrm{rank}(\Lambda_1) \leq r_1$. Update $\bar{W}^{k+1}$ as

$$\bar{W}^{k+1} \quad = \quad \frac{1}{\alpha^k + \mathrm{tr}(\Lambda)} (\alpha^k \bar{W}^k + P^k Q_2 \Lambda_2 (P^k Q_2)^T) \tag{2.33}$$

3. **Update P and hence X** Compute $\lambda_{min}(C - \mathcal{A}^T y^{k+1})$, and a corresponding eigenvector v^{k+1}. Also compute P^{k+1} as

$$P^{k+1} \quad = \quad \mathrm{orth}([P^k Q_1, v^{k+1}]) \tag{2.34}$$

   namely we take an orthonormal basis of $[P^k Q_1, v^{k+1}]$.

4. **Termination :** If

$$f(x^k) - \hat{f}^k(y^{k+1}) \quad \leq \quad \epsilon$$

   then **stop**

5. **Serious step :** If

$$f(y^{k+1}) \quad \leq \quad f(x^k) - m_L(f(x^k) - \hat{f}^k(y^{k+1}))$$

   set $x^{k+1} = y^{k+1}$, go to step 7.

6. **Null step :** Else $x^{k+1} = x^k$.

7. Increase k by 1, and go to step 2.

We have omitted the details regarding the convergence of the scheme. For more details refer to Helmberg et al [45, 52].

The spectral bundle scheme has very good global convergence properties. However it is only a *first order* scheme, since we are carrying out a first order approximation of the minimum eigenvalue function. A second order bundle method which converges globally and which enjoys asymptotically a quadratic rate of convergence was developed by Oustry [93]. Helmberg and Kiwiel [50] also extend the spectral bundle approach to problems with bounds.

2.7 A set of linear constraints

A set of linear constraints for (LDR) can be derived from the bundle information used by the spectral bundle method. We propose using the columns of P as the vectors $\{d_j\}, j = 1, \ldots, r$ in (LDR). Since the number of vectors r in the bundle P is $O(\sqrt{k})$ and we need at least k constraints to guarantee a basic feasible solution, we need to look for other constraints as well. We shall henceforth label these constraints as *box* constraints. Note that the columns of P are dense, leading to a dense linear programming formulation (LDR). We try to compensate for these dense constraints, by choosing d for our box constraints that are sparse.

This section is organized as follows. The rationale for using the columns of P as d is discussed in section 2.7.1. We illustrate the LP procedure on the max cut problem in section 2.7.2, and the min bisection problem in section 2.7.3.

2.7.1 Rationale for using the columns of P

The spectral bundle method returns a primal matrix X which is a convex combination of the aggregate matrix $\bar{W}$, and the columns of the bundle matrix P, i.e. $X = \alpha\bar{W} + PVP^T$, with $\alpha + \mathrm{tr}(V) = 1$, $\alpha > 0$, and $V \succeq 0$. Now consider

$$\begin{aligned} X &= \alpha\bar{W} + PVP^T \\ &= \bar{P}\bar{V}\bar{P}^T \end{aligned} \tag{2.35}$$

We are now ready to present the primary result in this chapter.

Theorem 19 *If the spectral bundle terminates, i.e.* $0 \in \partial\lambda_{min}(S)$*, then columns in* $\bar{P}$ *are the desired cutting planes* d_i *in the LP approach.*

Proof: First we note that $X = \alpha\bar{W} + PVP^T$ is a positive semidefinite matrix. This follows from $\bar{W}$ (see the update rule (2.33)), and $p_i p_i^T$, $i =, 1 \ldots, n$ being psd matrices, and hence X being a convex combination of these is psd too. Also, we assumed at the outset that the spectral bundle terminates in a finite number of steps. This will happen when $0 \in \partial\lambda_{min}(S)$. This implies that $d = 0$ in $(QSDP_{sb})$, since

$$\begin{aligned} 0 \in \partial\lambda_{min}(S) &\Rightarrow \lambda'_{min}(y, D) = 0 \\ &\Rightarrow \quad d = 0 \end{aligned}$$

From (2.32), we then have $\mathcal{A}(X) = b$, i.e. X is feasible in (SDP). Thus X is the optimal solution to (SDP). Now consider a spectral decomposition $X = \bar{P}\bar{V}\bar{P}^T$ as in (2.35). Since we have strong duality (Assumption 1), the vectors in $\bar{P}$ provide a basis for the null space of S (in fact they constitute the entire subdifferential for $\lambda_{min}(S) = 0$). Thus, from theorem 17 we find the columns of $\bar{P}$ ensure $\text{val}(LDR) = \text{val}(SDP) = \text{val}(SDD)$. □

Typically the bundle method converges only in the limit to the optimal solution. Thus the current X is not quite primal feasible, since we do not really have $\mathcal{A}(X) = b$. Moreover, we have

$$\begin{aligned} \text{tr}(X) &= \text{tr}(\alpha\bar{W} + PVP^T) \\ &= \alpha + \text{tr}(V) \\ &= 1 \\ &\approx \text{tr}(V) \end{aligned}$$

The second equality follows from the fact that $\text{tr}(\bar{W}) = 1$ (2.33), and $\text{tr}(PVP^T) = \text{tr}(VP^TP) = \text{tr}(V)$ (since $P^TP = I_r$). The last approximation follows from the fact that P contains the important subgradient information. Note that if we had exactly r columns, as given by Theorem 16 in P, then $\alpha = 0$. The following discussion indicates that using the columns of P (rather than $\bar{P}$) alone should suffice. We shall return to this in section 2.9.

2.7.2 The Max Cut problem

A semidefinite programming relaxation of the max cut problem is Goemans and Williamson [36]. The SDP solution followed by a randomized rounding leads to an 0.878 approximation algorithm for the max cut problem.

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{subject to} & \text{diag}(X) & = & e \\ & X & \succeq & 0, \end{array} \tag{2.36}$$

with dual

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{subject to} & -\text{Diag}(y) + S & = & -\frac{L}{4} \\ & S & \succeq & 0 \end{array} \qquad (2.37)$$

Here $L = \text{Diag}(Ae) - A$ is the Laplacian matrix of the graph, where A is the weighted adjacency matrix with $A_{ii} = 0$, $\forall i$ and $A_{ij} = w_{ij}$, $\forall \{i,j\} \in E$. Thus the Laplacian matrix is

$$\begin{array}{rcll} L_{ii} & = & \sum_j w_{ij} & \forall i \\ L_{ij} & = & -w_{ij} & i \neq j \end{array}$$

Note that the a in $\text{tr}X = a$ is trivially n, the number of nodes in the graph.

Since S is psd, we have $d^T S d = d^T(\text{Diag}(y) - \frac{L}{4})d \geq 0$, $\forall d$. In particular we propose to use the following d for the max cut problem.

MC1 Setting $d = e_i$, $i = 1 \ldots n$, where e_i is the ith standard basis vector for R^n. In particular e_i has a one in the ith position and zeros elsewhere. This generates the constraint $y_i \geq \frac{L_{ii}}{4}$, $i = 1 \ldots n$.

MC2 Setting $d = (e_i + e_j)$ and $(e_i - e_j)$, $\forall \{i,j\} \in E$, gives rise to the constraints $y_i + y_j \geq \frac{Lii}{4} + \frac{Ljj}{4} + \frac{L_{ij}}{2}$ and $y_i + y_j \geq \frac{Lii}{4} + \frac{Ljj}{4} - \frac{Lij}{2}$ respectively. Together these give $y_i + y_j \geq \frac{Lii}{4} + \frac{L_{jj}}{4} + |\frac{Lij}{2}|$.

MC3 The constraints in the bundle namely the columns p_i , $i = 1, \ldots, r$ of the matrix P.

We consider an LP relaxation $(LP1)$ containing MC1 and MC3. To obtain tighter relaxations we consider another relaxation $(LP2)$ containing MC2, in addition to MC1 and MC3.

The first relaxation is :

$$\begin{array}{lrcll} \min & e^T y & & & \\ \text{subject to} & y_i & \geq & \frac{L_{ii}}{4} \;\; i = 1, \ldots, n & (LP1) \\ & \sum_{i=1}^n p_{ji}^2 y_i & \geq & p_j^T \frac{L}{4} p_j, \;\; j = 1, \ldots, r & \end{array}$$

with dual

$$\begin{array}{lll} \max & \sum_{i=1}^{n} \frac{L_{ii}}{4} x_i + \sum_{j=1}^{r} w_j \frac{p_j^T L p_j}{4} & \\ \text{subject to} & \begin{bmatrix} 1 & & & \uparrow & & \uparrow \\ & \ddots & & p_1^2 & \ldots & p_r^2 \\ & & 1 & \downarrow & & \downarrow \end{bmatrix} \begin{bmatrix} x \\ w \end{bmatrix} = e & (LD1) \\ & \begin{bmatrix} x \\ w \end{bmatrix} \geq 0 & \end{array}$$

Here p_{ji} refers to the jth component of vector p_i and p_i^2, $i = 1, \ldots, r$ are vectors obtained by squaring all the components of p_i , $i = 1, \ldots, r$. $(LP1)$ has $n + r$ constraints in all. Note that $x \in I\!R^n$ and $w \in I\!R^r$ are the dual variables corresponding to $y \geq \text{diag}\frac{L}{4}$ and the bundle constraints respectively. To get a solution X to SDP, set $X = \text{Diag}(x) + \sum_{j=1}^{r} w_j p_j p_j^T$. This matrix X is positive semidefinite since $x \geq 0$ and $w \geq 0$. Moreover

$$\begin{array}{rcl} \frac{L}{4} \bullet X & = & \frac{L}{4} \bullet (\text{Diag}(x) + \sum_{j=1}^{r} w_j p_j p_j^T) \\ & = & \sum_{i=1}^{n} \frac{L_{ii}}{4} x_i + \sum_{j=1}^{r} w_j \frac{p_j^T L P_j}{4} \end{array}$$

This is precisely the objective value in $(LD1)$. We have thus generated the X which could be used in the Goemans and Williamson rounding procedure [36] to generate an approximate solution to the max cut problem.

Using the Goemans and Williamson [36] (GW) rounding procedure on the X generated by solving the relaxation LP, we can generate a cut that is at least 0.878 times the LP objective value. We cannot guarantee that the objective value of relaxation LP is an upper bound on the maximum cut value. However, in practice the LP objective is within 1% of the spectral bundle objective value, which incidentally is an upper bound on the optimal SDP value. Thus we have some performance guarantee on the cut produced by solving the LP relaxation LP followed by the GW randomized rounding procedure.

The LP relaxation $(LP2)$ is (we present only the primal here)

$$\begin{array}{lrcll} \min & e^T y & & & \\ \text{subject to} & y_i & \geq & \frac{L_{ii}}{4}, \ i = 1, \ldots, n & \\ & y_i + y_j & \geq & \frac{L_{ii}}{4} + \frac{L_{jj}}{4} + |\frac{L_{ij}}{2}| \ \forall \{i,j\} \in E & (LP2) \\ & \sum_{i=1}^n p_{ji}^2 y_i & \geq & p_j^T \frac{L}{4} p_j, \ j = 1, \ldots, r & \end{array}$$

Using the same discussion as above, we can generate a primal feasible matrix X for (SDP).

2.7.3 The Min Bisection problem

The semidefinite programming relaxation for the min bisection problem was independently proposed by Frieze and Jerrum [32] and Ye [120]. The relaxation is

$$\begin{array}{lrcl} \min & \frac{L}{4} \bullet X & & \\ \text{subject to} & \text{diag}(X) & = & e \\ & ee^T \bullet X & = & 0 \\ & X & \succeq & 0, \end{array} \tag{2.38}$$

with dual

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{subject to} & -y_0(ee^T) - \text{Diag}(y) + S & = & \frac{L}{4} \\ & S & \succeq & 0 \end{array} \tag{2.39}$$

We must note that Frieze and Jerrum [32] had the equipartition constraint as $ee^T \bullet X \leq 0$. But since the optimal X is psd, we must have $e^T X e = ee^T \bullet X \geq 0$ at optimality, which is equivalent to $ee^T \bullet X = 0$.

Here L refers to the Laplacian matrix of the graph. y_0 is the dual variable corresponding to the constraint $ee^T \bullet X = 0$. To get the signs right, we need to take the negative of the objective value of (SDD) to get the optimal solution to the min bisection problem. Again $a = n$.

Note that the primal (2.38) does not have any Slater point (strictly feasible

point) due to the equipartition constraint $ee^T \bullet X = 0$. Thus (SDD) need not attain its optimal solution. (There is no duality gap however since the dual (2.39) has a Slater point). Moreover (2.39) has an unbounded optimal face. To observe this set $y_0 \to \infty$ in (2.39). Doing so keeps S psd, but since y_0 does not appear in the objective function, this value remains unchanged.

In fact if we rewrite the equipartition constraint as $ee^T \bullet X = u$, the following can occur based on the values of u.

1. For $0 < u < n^2$, the primal problem is strictly feasible, and the dual attains its optimal solution.

2. For $u \in \{0, n^2\}$, strong duality holds since the dual has a Slater point, but the primal has no Slater point. To see this, for $u = 0$ equipartition constraint $e^T X e = 0$ implies that X is singular, in case $u = n^2$ the only primal feasible $X = ee^T$ which is singular (since the constraints $X \succeq 0$, and $\mathrm{diag}(X) = e$, automatically enforce $-1 \leq X_{ij} \leq 1$, $\forall i \neq j$). For $u = 0$ we get the equipartition constraint.

3. Finally, for $u < 0$, or $u > n^2$, the primal problem is infeasible. For the case $u < 0$, this implies $e^T X e < 0$ which is impossible (since X is psd). Also, for $u > n^2$, we have a infeasible problem, since $-1 \leq X_{ij} \leq 1$, $\forall i \neq j$.

We must emphasize here, that the absence of a primal Slater point affects our LP relaxations (since we are dealing with (LDD)), and we need not have a discretization, if (LDD) does not attain its optimal solution (Theorem 11).

Since $S = y_0 ee^T + \mathrm{Diag}(y) + \frac{L}{4}$ is psd, we require $d^T S d = d^T (y_0 ee^T + \mathrm{Diag}(y) + \frac{L}{4}) d \geq 0$, $\forall d$.

In particular we propose to use the following d for the min bisection problem.

MB1 Setting $d = e_i$, $i = 1, \ldots, n$ gives $y_0 + y_i \geq -\frac{L_{ii}}{4}$, $i = 1, \ldots, n$.

MB2 Setting $d = e$, where e is the all ones vector gives $n y_0 + \sum_{i=1}^n y_i \geq 0$, since $Le = 0$.

MB3 The constraints in the bundle namely the columns p_i, $i = 1, \ldots, r$. This gives $y_0 (p_i^T e)^2 + \sum_{j=1}^n p_{ji}^2 y_i \geq -p_i^T \frac{L}{4} p_i$, $i = 1, \ldots, r$.

MB4 Since the SDP has an unbounded feasible set, we impose an upper bound on y_0 say u.

The resulting LP is

$$\begin{array}{lrcll} \min & e^T y & & & \\ \text{subject to} & y_0 + y_i & \geq & -\frac{Lii}{4} & i = 1, \ldots, n \\ & n y_0 + \sum_{i=1}^n y_i & \geq & 0 & \\ & (p_i^T e)^2 y_0 + \sum_{j=1}^n p_{ji}^2 y_i & \geq & -p_i^T \frac{L}{4} p_i & i = 1, \ldots, r \\ & y_0 & \leq & u & \end{array} \tag{2.40}$$

Here p_{ji} refers to the jth component of the vector p_i. The LP (2.40) has $n + 1 + r$ constraints in all (excluding the upper bound).

If we set the upper bound u, we are in essence solving the following pairs of SDP.

$$\begin{array}{lrcl} \min & \begin{bmatrix} \frac{L}{4} & 0 \\ 0 & u \end{bmatrix} & \bullet & \begin{bmatrix} X & 0 \\ 0 & x_s \end{bmatrix} \\ \text{subject to} & \text{diag}(X) & = & e \\ & e e^T \bullet X & = & x_s \\ & \begin{bmatrix} X & 0 \\ 0 & x_s \end{bmatrix} & \succeq & 0 \end{array} \tag{2.41}$$

with dual

$$\begin{array}{lrcl} \max & 0 y_0 + e^T y & & \\ \text{subject to} & \text{Diag}(y) + e e^T y_0 + S & = & \frac{L}{4} \\ & S & \succeq & 0 \\ & y_0 & \leq & u \end{array} \tag{2.42}$$

Here x_s is the dual variable corresponding to the upper bound constraint $y_0 \leq u$. Also we have $e e^T \bullet X = x_s$ in (2.41). Similarly the dual variable corresponding to this upper bound constraint in (2.40) should provide an estimate for $e e^T \bullet X$. Note that the (2.41) has a Slater point and hence the dual (2.42) attains its optimal

solution.

2.8 Computational results

In this section we test the linear programming approach on the max cut and min bisection problems. The instances are taken from the 7th DIMACS Implementation Challenge [102] and Borchers' SDPLIB [17]. The bundle constraints are computed using Helmberg's spectral bundle code *SBmethod, Version 1.1* [48] available at *http://www.zib.de/helmberg/index.html*. *CPLEX 6.5* [25] is employed in solving the LP relaxations. All tests are executed on a *Sun Ultra 5.6, 440MHz* machine with *128* MB of memory.

We utilize the default bundle parameters which are :

1. The relative tolerance (*-te*). The default value is $1e-5$.
2. The size of the bundle i.e. the number of columns in P. This in turn is controlled by
   (a) The maximum number of vectors kept n_k (*-mk*). The default value is 20.
   (b) The maximum number of vectors added (*-ma*). The default value is 5.
   (c) The minimum number of vectors added n_{min} (*-mik*). The default value is 5.

The columns in the tables represent

n Number of nodes in the graph.

k Number of SDP constraints.

m Number of edges in the graph.

r Bundle size, the number of columns in P.

% Error $|\frac{SDP-LP}{SDP} \times 100|$.

SDP The objective value of SDP.

LP The value of LP relaxation.

m1 The number of constraints in the LP relaxation.

Name	n	m	r	SDP	LP	% Error
toruspm-8-50 [1]	512	1536	16	527.81	525.91	0.36
toruspm3-15-50 [1]	3375	10125	21	3.47e+03	3.43e+03	1.15
torusg3-8 [1]	512	1536	13	4.57e+07	4.54e+07	0.66
torusg3-15 [1]	3375	10125	18	3.13e+08	3.10e+08	0.96
mcp100 [2]	100	269	10	226.16	225.75	0.18
mcp124-1 [2]	124	149	20	141.99	141.06	0.65
mcp124-2 [2]	124	318	11	269.88	268.97	0.34
mcp124-3 [2]	124	620	11	467.75	467.37	0.08
mcp124-4 [2]	124	1271	10	864.41	863.72	0.08
mcp250-1 [2]	250	331	10	317.26	317.18	0.03
mcp250-2 [2]	250	612	14	531.93	531.18	0.14
mcp250-3 [2]	250	1283	13	981.17	980.32	0.09
mcp250-4 [2]	250	2421	13	1681.96	1679.70	0.13
mcp500-1 [2]	500	625	10	598.15	594.12	0.67
mcp500-2 [2]	500	1223	13	1070.06	1069.90	0.02
mcp500-3 [2]	500	2355	14	1847.97	1843.20	0.26
mcp500-4 [2]	500	5120	15	3566.74	3559.80	0.19
maxG11 [2]	800	1600	11	629.16	625.99	0.50
maxG32 [2]	2000	4000	14	1567.64	1557.91	0.62
maxG51 [2]	1000	5909	19	4003.81	3988.30	0.39

Table 2.1: Max Cut Test Results

In table 2.1 we compare the SDP objective value with the value of the LP relaxation. It is seen that the LP relaxation provides a fairly good approximation to the SDP objective value, with the $\%error$ within a % of the SDP objective value. We list the sizes of the LP relaxation in table 2.2. Note that $k = n$, for the max cut problem. Thus the LP relaxation has $(n + \sqrt{n}) = O(n)$ constraints.

We present the strength of the various LP relaxations, and the times involved in tables (2.3) and (2.4) respectively. The various LP relaxations are :

1. $LP1$ **:** This is $(LP1)$.

2. $LP2$ **:** This is $(LP2)$.

[1] DIMACS [102]
[2] SDPLIB [17]
[3] Run out of memory

Name	SDP		LP
	k	n	m1
toruspm-8-50	512	512	528
toruspm3-15-50	3375	3375	3396
torusg3-8	512	512	525
torusg3-15	3375	3375	3393
mcp100	100	100	110
mcp124-1	124	124	144
mcp124-2	124	124	135
mcp124-3	124	124	135
mcp124-4	124	124	134
mcp250-1	250	250	260
mcp250-2	250	250	264
mcp250-3	250	250	263
mcp250-4	250	250	263
mcp500-1	500	500	510
mcp500-2	500	500	513
mcp500-3	500	500	514
mcp500-4	500	500	515
maxG11	800	800	811
maxG32	2000	2000	2014
maxG51	1000	1000	1019
maxG55	5000	5000	5025
maxG60	7000	7000	7025

Table 2.2: Sizes of the max cut relaxations

3. $LP3$ **:** This is $(LP1)$, but with the columns of P replaced by the columns of $\bar{P}$ (see section 2.7.1 for the definition of $\bar{P}$).

4. $LP4$ **:** This is $(LP2)$, but with the columns of P replaced by the columns of $\bar{P}$.

The spectral bundle provides an upper bound on the SDP value (since we are solving a restricted version of the dual which is a minimization problem), while our LP relaxations provide lower bounds (since we are solving a relaxation of this SDP). As seen from table (2.3) these upper and lower bounds are fairly tight. Also, there is nothing to be gained by considering a spectral decomposition $X = \bar{P}\bar{V}\bar{P}^T = (\alpha\bar{W}+PVP^T)$, besides this matrix X is dense, and a spectral decomposition is costly.

Name	m	n	r	SDP	Bundle	LP1	LP2	LP3	LP4
gpp100	264	100	11	221.78	221.78	221.69	221.71	221.69	221.71
gpp124-1	149	124	20	141.91	141.99	141.06	141.37	141.06	141.37
gpp124-2	318	124	11	269.64	269.88	268.97	269.21	268.97	269.21
gpp124-3	620	124	11	467.68	467.75	467.37	467.44	467.37	467.44
gpp124-4	1271	124	10	864.26	864.42	863.72	863.81	863.72	863.81
gpp250-1	331	250	10	317.26	317.27	317.26	317.26	317.26	317.26
gpp250-2	612	250	14	531.78	531.93	531.18	531.38	531.18	531.38
gpp250-3	1283	250	13	980.91	981.18	980.32	980.48	980.32	980.48
gpp250-4	2421	250	13	1681.96	1681.96	1679.70	1680.01	1679.70	1680.02
gpp500-1	625	500	10	598.11	598.15	594.12	596.89	594.13	596.90
gpp500-2	1223	500	13	1070.00	1070.07	1069.91	1069.95	1069.91	1069.95
gpp500-3	2355	500	14	1846.60	1847.99	1843.24	1844.13	1843.24	1844.13
gpp500-4	5120	500	15	3566.74	3566.75	3559.79	3560.64	3559.8	3560.64

Table 2.3: Comparison of the various relaxations for maxcut

Hence we will disregard approaches $LP3$, and $LP4$, and present computational times only for the first two approaches. These can be found in table (2.4).

The spectral bundle is fairly erratic on this problem set. It takes it more than 6 hours to converge for problem gpp500-1; this is probably due to a bad choice of bundle parameters (default in this case). On an average, the bundle method is extremely quick. On some of the more dense problems, the time taken in our approach is dominated by solving the relaxation ($LP2$), which incidentally has $O(m+n)$ constraints. This could be as large as $O(n^2)$ for a complete graph.

In table 2.5 we compare the SDP objective value with the value of the LP relaxation (2.40) of the min bisection problem. Here $u = 1$ is the upper bound on the variable y_0. It is seen that the LP relaxation provides a good approximation to the SDP objective value. Moreover the dual variable x_s provides an estimate for $|ee^T \bullet X|$. This value is well below 0.1 for all the reported instances. A typical LP relaxation has $n + 1 + \sqrt{n} = O(n)$ constraints.

2.9 Conclusions

In this chapter we have described an LP approach to solving SDP problems. The LP approach uses the columns of P, in the spectral bundle approach developed

Problem Name	Bundle Time	LP1 Time	LP2 Time
gpp100	6	0.1	0.4
gpp124-1	106	0.2	0.6
gpp124-2	4	0.1	0.5
gpp124-3	5	0.1	2
gpp124-4	10	0.1	14
gpp250-1	3520	0.1	0.4
gpp250-2	20	0.1	1.5
gpp250-3	25	0.1	11
gpp250-4	20	0.2	79
gpp500-1	22248	0.5	1
gpp500-2	596	0.6	13
gpp500-3	38	0.7	112
gpp500-4	49	0.7	810

Table 2.4: The various times involved

Name	m	n	r	SDP	LP	m1	% Error	$\lvert ee^T \bullet X \rvert$
bm1 [1]	4711	882	10	23.44	24.99	893	6.59	0.06
gpp100 [2]	264	100	10	44.94	45.85	111	0.44	0.00
gpp124-1 [2]	149	124	10	7.34	7.35	135	0.01	0.00
gpp124-2 [2]	318	124	10	46.86	47.52	135	0.55	0.00
gpp124-3 [3]	620	124	11	153.01	153.59	136	0.16	0.00
gpp124-4 [2]	1271	124	11	418.99	420.35	261	0.14	0.01
gpp250-1 [2]	331	250	10	15.45	15.45	261	0.88	0.00
gpp250-2 [2]	612	250	12	81.87	82.22	263	0.26	0.00
gpp250-3 [2]	1283	250	13	303.50	305.99	264	0.56	0.00
gpp250-4 [2]	2421	250	13	747.30	751.72	264	0.43	0.00
gpp500-1 [2]	625	500	11	25.30	26.83	512	3.62	0.00
gpp500-2 [2]	1223	500	13	156.06	158.75	514	1.06	0.00
gpp500-3 [2]	2355	500	15	513.02	520.21	516	0.95	0.01
gpp500-4 [2]	5120	500	15	1567.02	1578.94	516	0.57	0.00
biomedP [1]	629839	6514	-	33.60	MM [3]	-	-	-
industry2 [1]	798219	12637	-	65.61	MM [3]	-	-	-

Table 2.5: Min Bisection Test Results

by Helmberg and Rendl [52] as constraints. The number of these constraints is bounded by $O(\sqrt{k})$, where k is the number of constraints in the SDP. The resulting linear programs can be solved quickly and provide reasonably accurate solutions. The key idea in using these constraints is that they provide a good approximation to the null space of S at optimality (see Theorem 17). It must emphasized that the main computational task in this aproach is in estimating the columns of the bundle P. Solving the resulting LP is relatively trivial.

The LP relaxation we solve is a relaxation to the dual SDP (SDD). The objective value returned by the spectral bundle scheme is for a feasible y, where $S = C - \mathcal{A}^T y$ is psd. Thus when (SDP) is a minimization problem, the two schemes give upper and lower bounds respectively on the SDP objective value. Thus our approach provides another termination criteria for the bundle approach (close to optimality), and also an upper bound. For the max cut problem we cannot guarantee that our LP relaxation is an upper bound on the max cut value. However in all cases the LP objective value is within a small percentage of that of the bundle, which is an upper bound on the max cut SDP value.

We found using the columns of $\bar{P}$ does not substantially improve the LP relaxations for the max cut and the min bisection problems. However this choice may be important for an SDP with a large number of constraints, such as k equipartition and Lovasz theta problems (since we attempt to solve these problems with a small number of columns in P). We refer the reader to Krishnan and Mitchell [69] for more details. As a result, the aggregate matrix $\bar{W}$, which contains the remaining subgradient information, becomes important.

We have successfully tested the LP approach on max cut and min bisection problems. However we must confess to not having much success with the Lovasz theta problems (Krishnan and Mitchell [69]). The large number of constraints in the Lovasz theta problem in turn require larger bundle sizes. In this regard, it might be interesting to consider a *second order* bundle scheme (Oustry [93]) to obtain faster convergence.

We can look at the LP approach presented in this chapter also as a way of generating feasible primal iterates for the spectral bundle method. When the

spectral bundle terminates, the primal matrix X is not as yet primal feasible, namely $\mathcal{A}(X) \approx b$ (in fact equality is attained only in the limit). It is important to get hold of a primal feasible X in an SDP cutting plane approach for combinatorial optimization problems. We shall return to this discussion in Chapter 4, where we describe an SDP approach for the maxcut problem. There is also some discussion in Chapter 5.

The spectral bundle approach can be looked upon as a cutting plane approach to solving the SDP. We motivate this in Appendix B.

To conclude, we have described an LP approach for an SDP with a bounded feasible set, which uses the spectral bundle approach as an oracle to generate the relevant linear constraints. The aim is to incorporate this LP approach in an overall branch and cut approach to solving integer programming problems.

2.10 Acknowledgments

The author wishes to thank Christoph Helmberg for making his spectral bundle code available, and for useful discussions.

CHAPTER 3
Cutting plane LP approaches to the SDP

3.1 Abstract

The SDP is a convex, semi-infinite linear programming problem; it has a polynomial time separation oracle, and can in practice be solved via a cutting plane approach. We present an interior point cutting plane LP approach in this chapter. The scheme is similar in spirit to interior point cutting plane methods (Mitchell and Ramaswamy [84], ACCPM [37, 38]) for convex optimization, and can be shown to have polynomial time complexity. However to make the resulting method practical, several refinements are necessary; we discuss some of these refinements in this chapter. We provide attractive empirical evidence using the approach, on an SDP arising as a relaxation of the maxcut problem. We shall utilize this approach as a subroutine, to solving the SDP relaxations, arising in a cutting plane approach for the maxcut problem in the next chapter.

3.2 Introduction

We present a cutting plane LP approach to the SDP in this chapter. The approach is based on a semi-infinite formulation of the SDP we introduced in the previous chapter. To make the method practical however, several refinements are necessary. In particular, the cutting plane method requires solving the LP relaxations using an interior point method, since this results in better cutting planes. We experiment with various separation oracles generating deep cuts, and techniques to restart the LP relaxations with strictly feasible starting points. Other issues such as updating the lower bounds, and techniques to generate sparser constraints are also discussed.

Before we begin, it is useful to take stock of the following :

1. Cutting plane methods are very different from interior point methods for the SDP discussed in the opening chapter. The schemes are usually less efficient

for problems to which interior point methods apply, i.e. unlike interior point methods we won't be able to solve an SDP to any degree of accuracy. It is hard enough to get even a few digits of accuracy.

2. Typically, the SDP relaxations we consider arise as relaxations of various combinatorial optimization problems. Without loss of generality, let us assume that our problem is a maximization problem (examples include the maxcut problem). The SDP relaxation serves as an upper bound, and we are not interested in solving it down to optimality. However we would still like to get a good solution since

   - A good upper bound can result in an early termination of the cutting plane approach for the underlying combinatorial optimization problem.
   - We also round the SDP solution to generate integer solutions; incidentally they serve as lower bounds in the cutting plane approach. Again, a good solution to the SDP is more likely to generate a good integer solution.

   In this chapter, we investigate cutting plane approaches in this regard.

3. The cutting plane methods discussed are similar in spirit to analytic center cutting plane methods for convex optimization, which incidentally are useful in solving non-differentiable optimization problems, and are competitors to bundle methods for non-differentiable optimization (discussed in the previous chapter). So far, the major work has been to prove improved complexity bounds for cutting plane methods; to our knowledge no one has really looked at practical implementations to solving general convex programming problems (like the SDP). This chapter serves to fill in this requirement.

4. There is a lot of flexibility in cutting plane methods to exploit problem structure. In contrast, interior point methods rarely exploit problem structure. Most of this flexibility is through the oracle employed in the cutting plane approach (we define an oracle in section 3.3).

This chapter is organized as follows : section 3.3 presents an overview of cutting plane methods : we introduce terminology, and motivate various techniques to gen-

erating cutting planes, some musings on the worst case complexity of a cutting plane approach are discussed in section 3.4, in section 3.5 we present our interior point cutting plane approach for the SDP, we discuss the various ingredients of this cutting plane approach in the subsequent subsections : section 3.5.1 discusses updating the lower bounds in the approach, warm start strategies are mentioned in section 3.5.2, the issue of solving the LP relaxations cheaply in 3.5.3, the separation oracles in section 3.5.4, and generating sparse constraints in section 3.5.5. We discuss the cutting plane approach on the maxcut SDP in section 3.6, computational results are presented in section 3.7, and we conclude with some observations, and conclusions in sections 3.8 and 3.9 respectively. A glossary of all the notation in this chapter can be found in Appendix A.

3.3 An overview of cutting plane methods

In this section, we present a short overview of cutting plane linear programming approaches to the SDP. The goal of cutting plane methods is to find a point in a convex set X, or to determine that X is empty. If we are dealing with an optimization problem, rather than a feasibility problem, then the set X is taken as the set of optimal points for the problem, and our goal is to find an optimal point for the optimization problem at hand.

We do not have access to any description of the convex set X (such as the objective function and constraint functions in an underlying optimization problem), except through an *oracle*. When we query the oracle at a point x, the oracle returns the following information : it either tells us that $x \in X$ (in which case we are done), or it returns a separating hyperplane that separates x from the set X. This follows from the *Hahn-Banach* theorem for convex sets (Rockafellar [105], Renegar [104]).

The cuts returned by the oracle at a point $\bar{x} \notin X$ have the following form :

$$a^T x \quad \geq \quad a^T \bar{x} + \gamma \tag{3.1}$$

If $\gamma > 0$, the cut is *deep*; if $\gamma < 0$, the cut is *shallow*, and finally if $\gamma = 0$, the cut passes through $\bar{x}$, and we shall refer to this as a *central* cut. We shall also assume

that $||a||_2 = 1$, since dividing a and γ by $||a||_2$ defines the same cutting plane. Our intuition suggests that deep cuts are better, since they exclude a larger set of points from consideration.

We will also assume access to a first order oracle for the objective function which, given an input point returns the objective function value, and a subgradient of the function at this query point. For the SDP we consider, the objective is a linear function, so we only need to compute the gradient of this function.

Thus the two main ingredients are :

1. A separation oracle for the feasible domain of the convex set at hand. We will utilize this oracle to generate cuts whenever we are infeasible, i.e. not in the convex set. The primary focus in this case is to attain feasibility.

2. A first order oracle (as discussed above) for the objective funtion. We will utilize this oracle to generate cuts whenever feasible, i.e. the query point is in the convex set. Thus the focus is to strive for optimality.

Consider the semidefinite programming problem :

$$\begin{array}{lrcll} \min & C \bullet X & & & \\ \text{subject to} & \mathcal{A}(X) & = & b & (SDP) \\ & X & \succeq & 0, & \end{array}$$

with dual

$$\begin{array}{lrcrcll} \max & b^T y & & & & & \\ \text{subject to} & \mathcal{A}^T y & + & S & = & C & (SDD) \\ & & & S & \succeq & 0 & \end{array}$$

We will assume that the SDP satisfies Assumption 1, i.e. both (SDP) and (SDD) have strictly feasible points. As mentioned in the previous chapter, we will be working with (SDD). Firstly, note that $S \succeq 0$ (a closed convex set) is equivalent to $\lambda_{min}(S) \geq 0$, i.e. $\lambda_{max}(-S) \leq 0$. Thus (SDD) can also be written as

$$\begin{array}{lrcl} \max & b^T y & & \\ \text{s.t.} & \lambda_{max}(\mathcal{A}^T y - C) & \leq & 0 \end{array} \tag{3.2}$$

We have a convex programming formulation in (3.2) : we are minimizing a linear function subject to a convex constraint $\lambda_{max}(\mathcal{A}^T y - C) \leq 0$. The maximum eigenvalue function is a non-smooth function, whenever this eigenvalue has a multiplicity greater than one. Any subgradient to the maximum eigenvalue function is given by pp^T, where p is a normalized eigenvector corresponding to $\lambda_{max}(\mathcal{A}^T y - C)$. Thus we have

$$\lambda_{max}(\mathcal{A}^T z - C) \quad \geq \quad \lambda_{max}(\mathcal{A}^T y - C) + pp^T \bullet (\mathcal{A}^T z - C - \mathcal{A}^T y + C), \quad \forall z \tag{3.3}$$

Now given the query point y, we first check for feasibility, i.e. $\lambda_{max}(\mathcal{A}^T y - C) \leq 0$. If y is not feasible, then we can construct a cut

$$\lambda_{max}(\mathcal{A}^T y - C) + pp^T \bullet (\mathcal{A}^T z - C - \mathcal{A}^T y + C) \quad \leq \quad 0 \tag{3.4}$$

To motivate this consider (3.4) with the reversed inequality. This would imply $\lambda_{max}(\mathcal{A}^T z - C) > 0$ from (3.3), and so z also violates the convex constraint. It follows that any feasible z satisfies (3.4). Using the fact that p is a normalized eigenvector corresponding to $\lambda_{max}(\mathcal{A}^T y - C)$, we have

$$\begin{aligned} pp^T \bullet (\mathcal{A}^T y - C) &= p^T(\mathcal{A}^T y - C)p \\ &= \lambda_{max}(\mathcal{A}^T y - C) \end{aligned}$$

Thus we can rewrite (3.4) as

$$pp^T \bullet (C - \mathcal{A}^T z) \quad \geq \quad 0 \tag{3.5}$$

a valid cutting plane, i.e. is satisfied by all the feasible z. Also, since $\lambda_{max}(\mathcal{A}^T y - C) > 0$ in (3.4), this is a *deep* cutting plane. Now suppose the point y is feasible, then we can construct a cutting plane

$$b^T(z - y) \quad \geq \quad 0 \tag{3.6}$$

We shall refer to (3.6) as an objective cut. Here, we are cutting out the half-space

$\{z : b^T(z - y) < 0\}$ because we know that all such points have an objective value less than y, and hence cannot be optimal. Note that b is simply the gradient of the objective function.

Another characterization of $S \succeq 0 \in \mathcal{S}^n$ is $\lambda_i \geq 0,\ i = 1, \ldots, n$. Thus any eigenvector corresponding to a negative eigenvalue of S gives a valid cutting plane.

Since the most negative eigenvalue of a symmetric matrix, and its associated eigenvector can be computed in polynomial time, the SDP actually has a polynomial time separation oracle. We can thus use this oracle in conjunction with the ellipsoid algorithm to generate a polynomial time algorithm for the SDP (this follows from the equivalence of separation and optimization established in Grotschel et al [42]; more details can be found in section 3.4).

We now present an alternative way to characterize the oracle to be utilized in the cutting plane approach. This is based on the semi-infinite linear programming formulation of the SDP introduced in the previous chapter. We will utilize this characterization in section 3.5.4. Consider the semi-infinite LP formulation (LDD).

$$\begin{array}{lll} \max & b^T y & \\ \text{s.t.} & d^T(C - \mathcal{A}^T y)d \ \geq \ 0, & \forall d \in B \qquad (LDD) \end{array}$$

Here B is a compact set (say)

$$B \ = \ \{d : ||d||_2 = 1\},$$

The ith iteration of the cutting plane scheme with the oracle is then :

1. Given $B_{i-1} = \{d_1, \ldots, d_{n^{i-1}}\} \subset B$. Solve the current LP relaxation (LDR_i), with the d's in B_{i-1}, namely

$$\begin{array}{lll} \max & b^T y & \\ \text{s.t.} & d_j^T(C - \mathcal{A}^T y)d_i \ \geq \ 0, & i = 1, \ldots, n^{i-1} \qquad (LDR_i) \end{array}$$

   Let us label the solution to (LDR_i) as y^i.

2. If $d^T(C - \mathcal{A}^T y^i)d \geq 0,\ \forall d \in B$, then we are done. To verify this we solve

the following subproblem, which incidentally is our oracle in the cutting plane approach.

$$\min \quad \{d^T(C - \mathcal{A}^T y^i)d : d \in B\} \tag{3.7}$$

If the optimum solution to (3.7) is < 0 then determine a number of approximate solutions t^i, $i = 1, \ldots, r^i$, for the subproblem (3.7). Stop, if ${t^l}^T(C - \mathcal{A}^T y^i)t^l \geq 0$, $l = 1, \ldots, r^i$. otherwise choose B_i such that

$$B_i \quad \subset \quad B_{i-1} \cup \{t^1, \ldots, t^{r^i}\} \tag{3.8}$$

3. Return to step 1

For instance, if we choose $B = \{d : ||d||_2 = 1\}$, then our oracle (3.7) is

$$\min \quad \{d^T(C - \mathcal{A}^T y^i)d : ||d||_2 \leq 1\} \tag{3.9}$$

Note that the objective function in (3.9) is a non-convex quadratic function. Thus any solution to (3.9) should occur on the boundary of the trust region $\{d : ||d||_2 = 1\}$. Thus we can replace $\{d : ||d||_2 = 1\}$ with $\{d : ||d||_2 \leq 1\}$. But (3.9) is just the variational characterization of the minimum eigenvalue of $(C - \mathcal{A}^T y^i)$, and any solution d is its associated eigenvector. This can be easily verified by writing the KKT conditions for (3.9). Thus we recover the cutting plane (3.5).

A few points are now in order :

1. The second scheme we described above, is referred to as an *exchange method* used to solve semi-infinite LP's (see the survey paper by Hettich and Kortanek [55] : section 7.1). The notation *exchange algorithm* refers to the fact that in every step a number of constraints are added (Step 2), and some of the old constraints, belonging to set B_{i-1} may conceivably be deleted, i.e. an exchange of constraints takes place.

2. This is similar to column generation approaches utilized in mixed-integer programming, except that we are dealing with an infinite, rather than a combi-

natorial number of constraints.

3. The subproblem (3.7) involves minimizing a non-convex objective function, and is usually an *NP hard* problem. However, as we have seen in the 2 norm case, we have a polynomial time routine thanks to the variational characterization of the minimum eigenvalue function.

3.4 Some musings on the complexity of the cutting plane approach

It must be mentioned that the SDP was known to be solvable in polynomial time, much before the advent of interior point methods. In fact we can use polynomial time oracle for the SDP mentioned in the previous section in conjunction with the ellipsoid algorithm to actually solve the SDP in polynomial time. It is interesting to compare the worst case complexity of such a method, with that of interior point methods.

The ellipsoid algorithm (Ben Tal and Nemirovskii [14], Papadimitriou and Steiglitz [96], and Grotschel et al [42]) can solve a convex programming problem of size n with a separation oracle, in $O(n^2 \log(\frac{1}{\epsilon}))$ calls to this oracle. This can be bettered to $O(n^2(\log(\frac{1}{\epsilon}))^2)$ using a potential reduction cutting plane algorithm, and to $O(n \log(\frac{1}{\epsilon}))$ for a volumetric barrier algorithm [1] (see Mitchell [83] for more details).

Let us examine the arithmetic complexity of the oracle discussed in the previous section. Let us assume that our current iterate is $\bar{y} \in \mathcal{R}^k$.

1. We first have to compute the dual slack matrix $\bar{S} = (C - \sum_{i=1}^{k} \bar{y}_i A_i)$, where $C, \bar{S}$, and A_i, $i = 1, \ldots, k \quad \in \mathcal{S}^n$. This can be done in $O(kn^2)$ arithmetic operations.

2. We then compute $\lambda_{min}(\bar{S})$, and it associated eigenvector d. This can be done in $O(n^3)$ arithmetic operations using the QR algorithm for computing eigenvalues, and possibly in $O(n^2)$ operations using the Lanczos scheme, whenever S is sparse.

[1] These analysis are valid for central cuts only

3. If $\lambda_{min}(\bar{S}) \geq 0$, we are feasible, and therefore we cut based on the objective function. This involves computing the gradient of the linear function, and this can be done in $O(k)$ time.

4. On the other hand if $\lambda_{min}(\bar{S}) < 0$, then we are yet outside the SDP cone; we can now add the valid constraint $\sum_{i=1}^{k} y_i(d^T A_i d) \leq d^T C d$, which cuts off the current infeasible iterate $\bar{y}$. The coefficients of this constraint can be computed in $O(kn^2)$ arithmetic operations.

The entire oracle can be implemented in $O(n^3 + kn^2)$ time. Also we would need $O(k^2 \log(\frac{1}{\epsilon}))$ calls to this oracle, and the entire process can be carried out in $O(k^2 n^2 (k+n) \log(\frac{1}{\epsilon}))$ arithmetic operations with the ellipsoid algorithm. Noting that k could be as large as $O(n^2)$, we solve the entire SDP in $O(n^8 \log(\frac{1}{\epsilon}))$ arithmetic operations, i.e. in polynomial time. This can reduced to $O(n^6 \log(\frac{1}{\epsilon}))$ if we use the volumetric barrier algorithm (however the cuts the returned by the oracle are no longer central; we would have to weaken them to utilize the complexity bound).

Let us compare this with the interior point approach discussed in section 1.6. Interior point methods (see Todd [110] for more details) can solve an SDP of size n, to a precision ϵ, in $O(\sqrt{n} \log(\frac{1}{\epsilon}))$ iterations (this analysis is for a short step algorithm). As regards the complexity of an iteration :

1. We need $O(kn^3 + k^2 n^2)$ arithmetic operations to form the Schur matrix M (see section 1.6). This can be brought down to $O(kn^2 + k^2 n)$ if the constraint matrices A_i have special structures (such as rank one as in the maxcut problem).

2. We need $O(k^3)$ arithmetic operations to factorize the Schur matrix, and compute the search direction. Again, this number can be brought down if we employ iterative methods.

The overall scheme can thus be carried out in $O(k(n^3 + kn^2 + k^2)\sqrt{n}L)$ arithmetic operations. Again setting $k = O(n^2)$, the overall scheme can be carried out in $O(n^{6.5} \log(\frac{1}{\epsilon}))$. operations. (We could some partial updating strategies to facotrize

M and improve on this complexity). Thus we could in practice improve the complexity of solving an SDP using a cutting plane approach. A few points are now in order :

1. These musings are only of theoretical interest : Interior point methods are *proven* ways for solving SDP problems, whereas the ellipsoid algorithm, in practice performs only as well as its worst case complexity.

2. The complexity analysis is not valid for the practical cutting plane algorithm we discuss in the rest of this chapter. Firstly, the complexity analysis of the volumetric barrier algorithm is only for central cuts, not for deep cuts we consider in our approach Also, some of the oracles we consider no longer have polynomial complexity.

3. However unlike interior point methods, we do exploit the structure of the underlying SDP problem by using problem specific oracles. Also, we are better able to exploit the sparsity of the problem.

3.5 An interior point cutting plane LP approach to the SDP

In this section we present a cutting plane linear programming approach for the SDP. As we have mentioned in the previous section, the SDP has a polynomial time separation oracle. Thus we can use the ellipsoid algorithm or interior point cutting plane algorithms with central cuts in a overall polynomial time algorithm to solve the SDP. However to make the resulting algorithm competitive with interior point methods for the SDP several refinements are necessary. We discuss some of these details in this section.

Consider the dual semidefinite program (SDD).

$$\begin{array}{lrcll} \max & b^T y & & & \\ \text{s.t.} & S & = & C - \mathcal{A}^T y & (SDD) \\ & S & \succeq & 0 & \end{array}$$

In every iteration of the cutting plane scheme, our LP relaxations (LDR), are progressively better discretizations of the semi-infinite LP formulation (LDD).

These provide upper bounds on the SDP objective value.

$$\begin{array}{lll} \max & b^T y & \\ \text{s.t.} & \sum_{j=1}^{k} (d_i^T A_j d_i) y_i \;\leq\; d_i^T C d_i, & i = 1, \ldots, m \qquad (LDR) \end{array}$$

The basic interior point algorithm for the SDP is the following :

1. **Initialize** The feasible region $S \succeq 0$ is unbounded. To implement any cutting plane method we need a bounded feasible set. The idea is to enclose the feasible region in a large box, whose dimesions are determined by the tolerance we wish to solve the SDP relaxation. This is not entirely trivial. This gives our starting LP relaxation. We shall return to this issue when we discuss the maxcut problem in section 3.6, where we have an easy starting relaxation (not bounded) thanks to the problem structure.

2. **Restart** the current LP relaxation with a strictly feasible starting point.

3. **Solve the LP relaxations** cheaply, to a moderate tolerance β. The solution to the dual LP provides an upper bound on the SDP value. Reduce β.

4. **Call the SDP separation oracle** to find violated cutting planes. Bucket sort the violated inequalities by violation, and add a subset of constraints to the relaxation. Drop any constraints that no longer appear important.

5. **Update lower bounds :** In each iteration S is not quite psd. Perturb S slightly so that it is just psd, and update lower bounds.

6. **Check for termination :** If the difference between the upper and lower bounds is small, and the magnitude of the most negative eigenvalue of S is small, STOP.

7. **Loop :** return to step 2

Each of these points is a research problem in itself, and we will devote a subsection of this chapter for each. We describe one iteration of the cutting plane approach in figure (3.1). Each iteration in the interior point cutting plane approach contains the following three ingredients :

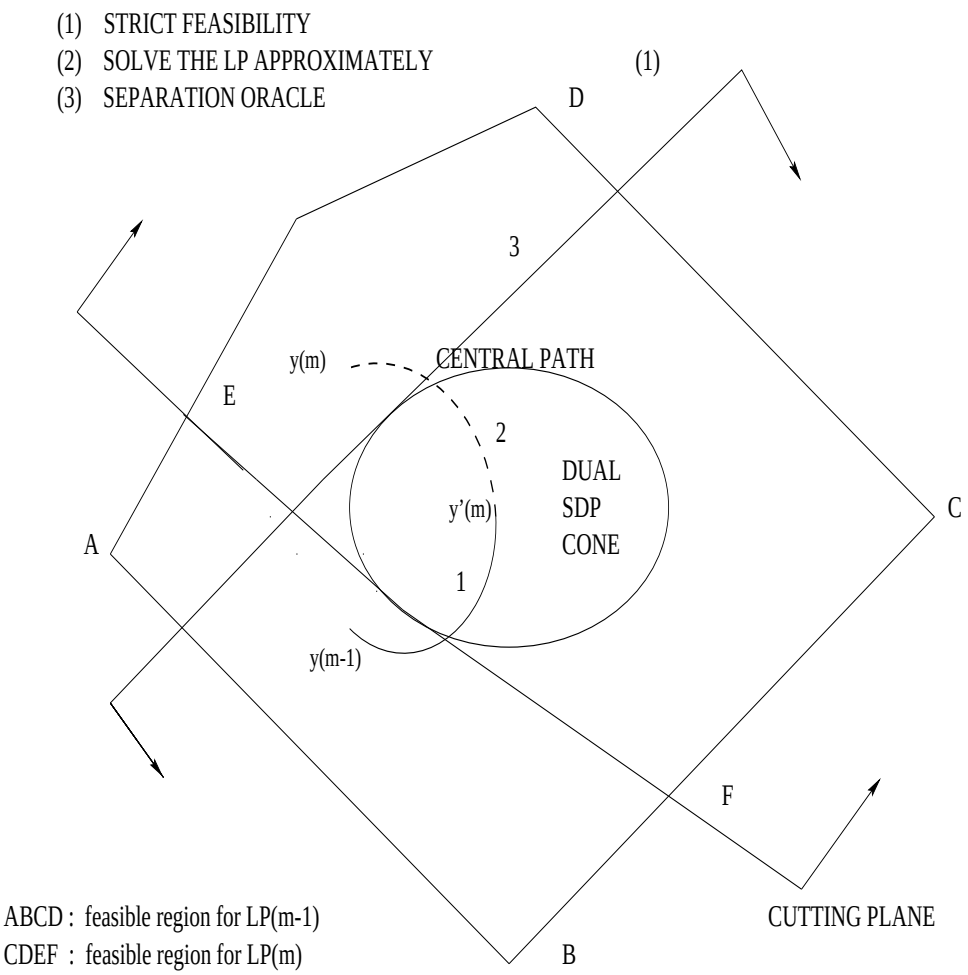


Figure 3.1: Solving the SDP via an LP cutting plane scheme : The three important stages in every iteration

1. **Restart with a strictly feasible starting point** : The oracle in the previous iteration returned a number of deep cuts, and the current iterate is no longer feasible. We first work towards restoring strict feasibility of the primal and dual iterates. We describe the procedure in section 3.5.2.

2. **Solve the LP relaxations cheaply using an interior point solver** : We solve the LP relaxations, with the strictly feasible starting point from step 1, approximately, gradually tightening the accuracy as needed. This generates better cutting planes. More details can be found in section 3.5.3.

3. **Call the oracle to generate deep cutting planes** : We experiment with two different oracles generating deep cutting planes. We also relate these

cutting planes to faces of the SDP cone. This can be found in section 3.5.4.

3.5.1 Updating lower bounds

The LP relaxations (LDR) provide upper bounds on the SDP objective value. We would like to estimate good lower bounds on the SDP objective value. Also, we would like this procedure to be cheap and dirty. These lower upper bounds serve as a termination criteria for the cutting plane approach, i.e. we stop when the difference between the lower and upper bounds is small; they will also be useful when we are solving combinatorial optimization problems via an SDP cutting plane approach, as discussed in the next chapter.

We formulate this problem, and mention a solution strategy below. Firstly, Assumption (2) ensures the existence of the following $\hat{y}$.

Theorem 20 *If every feasible X for (SDP) satisfies $tr(X) = a$, then*

$$\exists \hat{y} \in \mathcal{R}^k \quad with \quad \mathcal{A}^T \hat{y} = I.$$

Moreover this $\hat{y}$ satisfies $b^T \hat{y} = a$.

Proof: Since $\mathrm{tr}(X) = a$ is satisfied for every feasible X in (SDP), it can be expressed as a linear combination of the other primal constraints $A_i \bullet X = b_i$, $i = 1, \ldots, k$. Letting the components of $\hat{y}$ to be the coefficients in this linear combination we get the desired result. □

Also, see Helmberg [45] (Proposition 5.1.1) for more details, including the reverse implication.

In every iteration, we force S to positive semidefiniteness, i.e. try and drive $\lambda_{min}(S)$ to zero. At the end of an iteration, the matrix $\bar{S} = (C - \mathcal{A}^T \bar{y})$ is not quite psd (it is in fact an indefinite matrix). Also, $\lambda_{min}(\bar{S})$ can be written as the following SDP (3.10) (see the examples on the SDP in Chapter 1 for a derivation).

$$\begin{array}{lrcl} -\min & \lambda & & \\ \text{s.t.} & S & = & \bar{S} + \lambda I \\ & S & \succeq & 0 \end{array} \tag{3.10}$$

This suggests, that if we estimate $\lambda_{min}(\bar{S})$ (say λ for a shorthand notation), and carry out the perturbation $S = \bar{S} - \lambda I$, then the resulting matrix S is psd. Setting $S = (C - \mathcal{A}^T y)$, we estimate y from $\mathcal{A}^T(y - \bar{y}) = \lambda I$; such a y exists from Theorem 20. In fact, $y = \bar{y} + \lambda \hat{y}$. A lower bound on the SDP objective value is then given by $b^T y$ (note that the dual problem is a maximization problem, and y is within the dual SDP cone from (3.10)).

It is worth investigating how good these bounds are, and whether we can consider any enhancements. For instance, let us consider the dual maxcut SDP (4.15) in this regard. As a caveat, we must mention that in the maxcut problem the primal SDP is actually a maximization problem, so we are actually trying to estimate an upper bound in this case (this is however irrelevant to the discussion that follows). In this case $\mathcal{A}^T y = \text{Diag}(y)$. Thus, we only have freedom to change the diagonal entries of $S = \text{Diag}(y) - \frac{L}{4}$. We could utilize the above procedure to estimate an upper bound; note that we are perturbing all the components of y by the same amount (see 3.10). Since the objective value in the maxcut SDP is $e^T y$, our upper bound is a factor of $n\lambda$ more than the current lower bound $e^T \bar{y}$. This bound need not be good, if n is large, and λ is not close to zero.

We can do better in this case by utilizing information from the eigenvector v corresponding to λ. The following idea was mentioned to us by Parlett (see Parlett [97]). We outline the steps below :

1. Compute
$$g \;=\; \frac{1}{\sum_{i=1}^{n} v_i^4}$$
2. Augment the kth diagonal entry by $g\ |\lambda|\ v_k^2$.
3. Iterate, until λ is small enough.

The increase is $g\lambda(\sum_{i=1}^{n} v_i^2) = g\lambda$ (since the eigenvector v is normalized). Also, since $||v||_2 = 1$, we have $-1 \leq v_i \leq 1$, $i = 1, \ldots, n$. Thus

$$\begin{aligned} g\ \lambda \;&=\; \frac{1}{\sum_{i=1}^{n} v_i^4}\ \lambda \\ &\leq\; n\ \lambda \end{aligned}$$

The maximum value of $g = \frac{1}{\sum_{i=1}^{n} v_i^4}$ subject to $||v||_2 = 1$ is n, and is attained when all the components of v are the same. The last inequality follows from this observation.

3.5.2 Restart with a strictly feasible starting point

In this section we discuss ways of restarting the primal and the dual LP relaxations with strictly feasible starting points. Without loss of generality, we will assume that our LP relaxations have the following form :

$$\begin{array}{lrcl} \max & b^T y & & \\ \text{subject to} & A^T y & \leq & c \end{array} \tag{3.11}$$

with dual

$$\begin{array}{lrcl} \min & c^T x & & \\ \text{subject to} & Ax & = & b \\ & x & \geq & 0 \end{array} \tag{3.12}$$

Let us fix some notation here : Our current iterate is (x, y, s), where $s = (c - A^T y)$. Also, $x, s \in \mathcal{R}^m$. We assume that p deep cuts $\bar{A}^T y \leq \bar{c}$ have been added in the dual (3.12). The iterates in the new LP relaxation are $(\tilde{x}, \tilde{y}, \tilde{s})$, where $\tilde{x} = \begin{bmatrix} x \\ \bar{x} \end{bmatrix} \in \mathcal{R}^{m+p}$, and $\tilde{s} = \begin{bmatrix} s \\ \bar{s} \end{bmatrix} \in \mathcal{R}^{m+p}$ respectively. Also, $\tilde{c} = \begin{bmatrix} c \\ \bar{c} \end{bmatrix} \in \mathcal{R}^{m+p}$, and $\tilde{A} = \begin{bmatrix} A & \bar{A} \end{bmatrix} \in \mathcal{R}^{n \times (m+p)}$. Note that new components of $\tilde{s}$, namely $\bar{s}_i << 0$, since the p new inequalities $\bar{A}^T y \leq \bar{c}$, are seriously violated by the current iterate $\bar{y}$, i.e. $\bar{A}^T \bar{y} >> \bar{c}$, (for instance if we are considering the 2 norm, the magnitude of the violation can get as large as the most negative eigenvalue of the dual slack matrix). The new LP relaxations are :

$$\begin{array}{lrcl} \max & b^T y & & \\ \text{subject to} & A^T y & \leq & c \\ & \bar{A}^T y & \leq & \bar{c} \end{array} \tag{3.13}$$

with dual

$$\begin{array}{lrcl} \min & c^T x + \bar{c}^T \bar{x} & & \\ \text{subject to} & Ax + \bar{A}\bar{x} & = & b \\ & x & \geq & 0 \\ & \bar{x} & \geq & 0 \end{array} \tag{3.14}$$

We first note that restoring primal feasibility, i.e. generating a strictly feasible starting point for (3.13) is a difficult, since the cuts introduced are deep, and this problem is a significant perturbation of (3.11).

We will use the fact that the SDP cone is always within the feasible region of the sequence of LP relaxations to restart the primal. The procedure is enunciated below.

To generate a strictly feasible starting point for (3.13), we choose a $\tilde{y}$ such that the dual slack matrix $\tilde{S} = C - \mathcal{A}^T \tilde{y}$ is positive definite. To do this, we estimate the most negative eigenvalue λ of the current dual slack matrix $S = C - \mathcal{A}^T y$, and perform the following updates (see the discussion in section 3.5.1) :

$$\begin{aligned} \tilde{y} &= y + \alpha\lambda\hat{y} \\ \tilde{S} &= S - \alpha\lambda I \end{aligned}$$

where $\hat{y}$ satisfies Theorem 20, and I is the identity matrix of the appropriate size, for some $\alpha > 1$. This update will ensure that $\tilde{s} = \tilde{c} - \tilde{A}^T \tilde{y} > 0$.

Now consider the dual (3.14). Firstly, $\tilde{x} = \begin{bmatrix} x \\ 0 \end{bmatrix}$ is feasible in (3.14), but not strictly feasible. To restore strict feasibility we utilize a direction originally due to Mitchell and Todd [85] for central cuts. (Also, see Gondzio [40], and Goffin and Vial [37]). We motivate the main steps in the derivation of this direction below.

1. Our aim is to increase the new components $\bar{x}$ from zero.

2. Consider the previous iterate x. This iterate is strictly within the old feasible region, i.e. $Ax = b, x > 0$. Now consider a step Δx. We have the simple observation that $x + \Delta x > 0$, for all Δx, such that $||X^{-1}\Delta x|| \leq 1$. In fact we can

in practice replace the X^{-1} with D^{-1}, where D is one of the following scaling matrices X, S or $(XS^{-1})^{\frac{1}{2}}$. Thus the ellipsoidal neighborhood $||D^{-1}\Delta x|| \leq 1$ (also referred to as the *Dikin* ellipsoid) of x, is entirely within the old feasible region $\{x : Ax = b, x \geq 0\}$.

3. Motivated by 1, and 2, we solve the following problem to restore strict dual feasibility :

$$\begin{array}{lrcl} \max & \sum_{i=1}^{p} \log \bar{x}_i & & \\ \text{s.t.} & A\Delta x + \bar{A}\bar{x} & = & b, \\ & ||D^{-1}\Delta x|| & \leq & 1 \\ & \bar{x} & \geq & 0 \end{array} \tag{3.15}$$

   Since $\sum_{i=1}^{p} \log \bar{x}_i$ is $-\infty$ at $\bar{x} = 0$, the condition $\bar{x} > 0$ is automatically enforced. Also $A\Delta x + \bar{A}\bar{x} = b$ is the new primal feasibility constraint.

4. The solution to (3.15) is given by

$$\Delta x = -D^2 A^T (AD^2 A^T)^{-1} \bar{A}\bar{x}$$

   This can be easily obtained by writing down the KKT conditions for (3.15).

5. In the instance, that we add central cuts in (3.13) Goffin and Vial [37] determine $\bar{x}$ by solving a strictly convex quadratic subproblem. However, in our case the cuts are deep; hence we will arbitrarily assign $\bar{x} = e$, where $e \in \mathcal{R}^p$ is the all ones vector.

To conclude, here are the main steps towards restoring dual feasibility :

1. Given $x > 0$. We need a strictly positive solution $\bar{x} > 0$. Assign $\bar{x}$ to be the all ones vector.

2. Construct $\Delta x = -D^2 A^T (AD^2 A^T)^{-1} \bar{A}\bar{x}$. Here $D^2 = XS^{-1}$ is the current primal dual scaling matrix.

3. Compute the maximum feasible step-size in the primal space

$$\alpha_{max} \quad = \quad \min_{j:\Delta x_j<0} \frac{-x_j}{\Delta x_j}$$

4. The new point is then

$$\tilde{x} \quad = \quad \begin{pmatrix} x + 0.9995\alpha_{max}\Delta x \\ 0.9995\alpha_{max}\bar{x} \end{pmatrix}$$

These step sizes employed are similar to those in Gondzio [40].

3.5.3 Solving the LP relaxations cheaply

We solve the LP relaxations in the interior point cutting plane approach approximately, i.e. initially to weaker tolerances, gradually tightening the degree of accuracy. This results in better cutting planes than a simplex LP solver, where the query points to the oracle are extreme points of the LP feasible region.

A few points are in order :

1. Our algorithm employs a long step interior point algorithm to solve the LP relaxations. This is in contrast to the discussion in section 3.3, where we essentially have a short step algorithm. This is because we add central cuts based on the objective function whenever feasible. A long step algorithm is more attractive from a computational perspective (see Mitchell and Ramaswamy [84] for other long step algorithms for solving convex programming problems).

2. We have observed that a simplex implementation performs very poorly. This is because the optimal solution is an extreme point on the boundary of the feasible region, which results in poor cutting planes.

3. We adjust our tolerances based on $\lambda_{min}(S)$. In our cutting plane approach, we try and force this eigenvalue to zero. If the magnitude of $\lambda_{min}(S)$ is close to zero, we are nearly feasible, and hence reduce the tolerance β to which we solve the LP relaxations by a larger amount. The aim is to strive for optimality

when feasible, keeping in the spirit of long step methods. The parameter β controls the tradeoff between feasibility and optimality.

4. Mitchell [82] has employed a similar idea in cutting plane approaches for linear ordering and maxcut problems with great success.

3.5.4 Separation oracles generating deep cuts

We now discuss the oracles to be utilized within the cutting plane approach; we had motivated the notion of an oracle in section 3.3. We discuss some of their properties, the cutting planes they generate, and some implications of these for our cutting plane approach.

Our oracle in the cutting plane approach is (3.7).

$$\min \quad \{d^T(C - \mathcal{A}^T y^i)d : d \in B\}$$

Here B is a compact set, especially

$$\begin{aligned} B &= \{d : ||d||_2 \leq 1\}, \\ \text{or } B &= \{d : ||d||_\infty \leq 1\} \end{aligned}$$

These are commonly employed in trust region methods (Conn et al [24]). Let us first consider the 2 norm case. We had discussed this case in detail in section 3.3. A few points are now in order :

1. Until we are optimal, we are dealing with a non-convex quadratic objective function, since $S = (C - \mathcal{A}^T y)$ is not psd. Thus the solution to the subproblem must lie on the boundary of the trust region. Also, we can replace the requirement $d^T d \leq 1$ with $d^T d = 1$.

2. A solution to the subproblem is an eigenvector corresponding to the minimum eigenvalue of $S = (C - \mathcal{A}^T y)$. Thus, although we are dealing with a non-convex problem, we can solve it in polynomial time (see variational characterization of the minimum eigenvalue function of a symmetric matrix in Horn and Johnson [58]). An interesting survey can also be found in Section 7.2 of Conn et al [24].

If this minimum eigenvalue is nonnegative, we are done.

3. Our cutting plane is deep, since we can rewrite our cutting plane as

$$\begin{aligned} dd^T \bullet (C - \mathcal{A}^T y) &\geq 0 \\ &= dd^T \bullet (C - \mathcal{A}^T \bar{y}) + \gamma \end{aligned}$$

Here $\bar{y}$ is our current query point, d is the eigenvector corresponding to $\lambda_{min}(C - \mathcal{A}^T \bar{y})$. Also, $\gamma > 0$, since $dd^T \bullet (C - \mathcal{A}^T \bar{y}) = \lambda_{min}(C - \mathcal{A}^T \bar{y}) < 0$.

4. The separation oracle corresponds to $\lambda_{min}(S) \geq 0$. Since

$$\lambda_{min}(S) \geq 0 \quad \Leftrightarrow \quad \lambda_i(S) \geq 0, i = 1, \ldots, n$$

we note that any eigenvector corresponding to a negative eigenvalue of S gives a valid cutting plane, and we can add all of these in each iteration. However this results in a considerable increase in the number of constraints. Therefore we specify a limit p on the number of eigenvectors to be introduced in each iteration. These extremal eigenvalues are easily computed using the *Lanczos* scheme (Golub and Van Loan [39]).

5. The approach leads to dense constraints making the LP relaxations harder to solve using an interior point approach. We discuss some approaches to generating sparse constraints in section 3.5.5.

We can also relate the eigenvector corresponding to the most negative eigenvalue of S to faces of the dual SDP cone $\{y : S = (C - \mathcal{A}^T y \succeq 0\}$.

Theorem 21 *The constraint $dd^T \bullet S = d^T(C - \mathcal{A}^T y)d \geq 0$, where d is the eigenvector corresponding to the most negative eigenvalue of the dual slack matrix S, is a face of the dual SDP feasible region (in y space) $\{y \in \mathcal{R}^k | S = (C - \mathcal{A}^T y) \succeq 0\}$.*

Proof: Let $\bar{S} = C - \mathcal{A}^T \bar{y}$ be our current indefinite matrix. Let λ be the magnitude of the most negative eigenvalue of $\bar{S}$, and d its corresponding eigenvector. From Theorem 20 $S = \bar{S} + \lambda I$ is psd, and if we set $y = \bar{y} + \lambda \hat{y}$, then $S = C - \mathcal{A}^T y$. Since S is psd, its smallest eigenvalue is now zero, with eigenvector d. Thus we have

$d^T S d = 0$. The point y is on the supporting hyperplane $d^T S d \geq 0$. There is at least one feasible y which satisfies the new constraint $d^T S d \geq 0$ at equality. □

The dimension of the face $d^T S d \geq 0$ is related to the multiplicity of the most negative eigenvalue of $\bar{S}$, or the zero eigenvalue of S. Let r define the multiplicity of the most negative eigenvalue of the current indefinite $\bar{S}$.

Theorem 22 *The dimension of the face $d^T S d \geq 0$ is $k - r$.*

Proof: Since the most negative eigenvalue of $\bar{S}$ has multiplicity r, and since $\bar{S}$ is symmetric, the eigenspace (space of linearly independent eigenvectors) is a r dimensional subspace of $\mathcal{R}^k$. Let $p_i,\ i = 1, \ldots, r$ form an orthonormal basis for this eigenspace. In short this basis provides r linearly independent supporting hyperplanes, i.e. constraints holding at equality at the point y. These are of the form

$$p_i^T \mathcal{A}^T y p_i \ = \ \sum_{j=1}^k y_j (p_i^T A_j p_i) \ = \ p_i^T C p_i, \ \ i = 1, \ldots, r$$

Thus the dimension of this face $\leq k - r$.

We shall show that the dimension of the face $\geq k - r$ by exhibiting $k - r + 1$ affinely independent points on $dd^T \bullet S = 0$. Let our current point be $\bar{y}$, with $\bar{S} = C - \mathcal{A}^T \bar{y}$ as our indefinite matrix. Proceeding as in the previous theorem we find that $y = \bar{y} + \lambda \hat{y}$, where λ is the magnitude of the $\lambda_{min}(\bar{S})$, and $\hat{y}$ satisfying Theorem 20 ensure $S = C - \mathcal{A}^T y$ is psd, with $\lambda_{min}(S) = 0$. Since this eigenvalue has multiplicity r, we have $d_i^T S d_i = 0,\ i = 1, \ldots, r$ (where $d_1, \ldots, d_r$ is a basis for the eigenspace). We want other $\tilde{y}$ such that $S(\tilde{y}) \succeq 0$ satisfies $d_i^T S(\tilde{y}) d_i = 0$, $i = 1, \ldots, r$. Setting $\tilde{y} = y + \alpha \tilde{d}$ we require $d_i^T (\mathcal{A}^T \tilde{d}) d_i = 0$, $i = 1, \ldots, r$. Note that for α sufficiently small $S(\tilde{y})$ is psd. Now

$$\begin{aligned} d_i^T (\mathcal{A}^T \tilde{d}) d_i \ &= \ \sum_{j=1}^k \tilde{d}_j (d_i^T A_j d_i) \\ &= \ \ 0, \ \ i = 1, \ldots, r \end{aligned} \tag{3.16}$$

There are r equations, and k unknowns, and so the dimension of the null space of the coefficient matrix is $k - r$. Thus there are $k - r$ linearly independent directions $\tilde{d}$ giving $k - r + 1$ affinely independent points y and $\tilde{y_j},\ j = 1, \ldots, k - r$ on the hyperplane $d^T S d \geq 0$. Thus the dimension of the face must be at least $k - r$. □

Corollary 7 *If $k = 1$, then $d^T S d \geq 0$ defines a* facet *of the dual SDP cone in y space.*

This has important consequences in the rate of convergence of the cutting plane approach. In the beginning, the multiplicity of the most negative eigenvalue of S is one, and the cutting planes returned by the 2 norm oracle are facets of the dual SDP cone. As the algorithm proceeds, we try and force this most negative eigenvalue to zero. Thus the smallest eigenvalues of S tend to coalesce together (see Pataki [99]), thereby increasing the multiplicity of the smallest eigenvalue of S. Thereafter, the cutting planes returned by the oracle are no longer facets, but lower dimensional faces of the SDP cone in y space.

One way to overcome this difficulty would be to replace the linear constraint $d^T S d \geq 0$, with the semidefinite constraint $P^T S P \succeq 0$ (where $P \in \mathcal{R}^{n \times r}$ contains eigenvectors corresponding to the most negative eigenvalue of S, which has multiplicity r). Note that this is stronger than $p_i^T S p_i \geq 0$, $i = 1, \ldots, r$. The resulting problem is no longer an LP, but can be solved within an ACCPM framework that incorporates semidefinite cuts (see Oskoorouchi [92]).

Now let us consider the ∞ norm case. In this case our subproblem has the following form :

$$\begin{array}{ll} \min & d^T(C - \mathcal{A}^T y)d \\ \text{s.t.} & -1 \leq d_i \leq 1 \quad i = 1, \ldots, n \end{array} \tag{3.17}$$

A few remarks are now in order

1. Unlike the 2 norm, there is no easy way to compute the global minimizer of the non-convex function in (3.17). Any QP solver can only guarantee a local minimizer. We restart the QP solver with a number of random starting points to find a number of local minimizers.

2. Any solution to (3.17) has quite a few components equal to ± 1. This results in a cutting plane approach with a better rate of convergence. We will return to this in section 3.6 on the maxcut problem.

We wish to emphasize an important point here. Just because an algorithm is polynomial, it does not mean that it is an effective method in practice, nor does it mean that an exponential algorithm for a NP hard problem is bad except in the worst case. However, this theory plays an important role in developing worst case bounds in algorithms such as the ellipsoid algorithm (Grotschel et al [42]), Ben Tal and Nemirovskii [14]), ACCPM [37, 38], and other polynomial interior point algorithms (see Mitchell [83] for a survey), where we want the algorithm to work well no matter which cutting plane is returned by the oracle.

To illustrate this, consider the max cut problem on the following complete graph with 3 nodes. The weights associated with the three edges $\{1,2\}$, $\{1,3\}$ and $\{2,3\}$ are 1, 4 and 2 respectively. This corresponds to a Laplacian matrix $\frac{L}{4} = \begin{bmatrix} 1.25 & -0.25 & -1 \\ -0.25 & 0.75 & -0.5 \\ -1 & -0.5 & 1.5 \end{bmatrix}$. The optimal SDP objective value is 6 (incidentally this is also the maxcut value, and corresponds to putting node 1 on one side, and nodes 2 and 3 on the opposite side). The cutting plane algorithm, with the ∞ norm solves the SDP in the 1st iteration itself. In the first iteration, $y = \mathrm{diag}(\frac{L}{4})$, and hence the infinity norm subproblem is to minimize $d^T \begin{bmatrix} 0 & 0.25 & 1 \\ 0.25 & 0 & 0.5 \\ 1 & 0.5 & 0 \end{bmatrix} d$, subject to $-1 \leq d_i \leq 1$, $i = 1, \ldots, 3$. The optimal value to this subproblem is -2.5, and is attained for $d = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$. This gives the constraint $\sum_{i=1}^{3} y_i \geq 6$, and since we are minimizing $\sum_{i=1}^{3} y_i$, this gives us the optimal solution. On the other hand the cutting plane scheme with the 2 norm, takes about 7 iterations to find the optimal solution. Thus the reason the ∞ norm seems to work well is that d's returned by the oracle mimic incidence vectors of cuts, since the trust region forces most of the components of d to be between ± 1. Thus it is more useful to consider the ∞ norm with regard the maxcut value, despite the fact solving this problem is NP hard.

We conclude this section with a short note on dropping constraints that no longer appear important. This is important to keep the size of the LP relaxations

small. We follows two strategies :

1. We drop those constraints whose dual variables are small. Another criterion is to use the ratio of the dual variable to the slack in the constraint, and drop constraints for which the ratio is small.

2. Here is a more intriguing approach : Construct the primal matrix $X = \sum_{i=1}^{m} x_i d_i d_i^T$, where x_i is the dual variable corresponding to the constraint $d_i d_i^T \bullet S \geq 0$. From theorem 9, we know that X is feasible in (SDP). Compute a spectral factorization $X = P\Lambda P^T$, and replace the vectors d_i, by the eigenvectors p_i, $i = 1, \ldots, r$ corresponding to nonzero eigenvalues of X (here r is the rank of X. Also, we know that $r = O(\sqrt{k})$, from theorem 16). The p_i encapsulate the important information contained in the d_i. It is interesting to relate this to the *aggregation* procedure employed in the spectral bundle approach of the previous chapter. We repeat the procedure periodically, to keep the size of the LP relaxations small.

3.5.5 Generating sparser constraints

We first mention the positive semidefinite matrix completion problem (see Laurent [73], and Grone et al [41]), and show that it can be utilized to generate sparse constraints while solving relaxations of (SDP) (discussed in the previous chapter). Fukuda et al [34] have utilized this idea within the context of interior point methods for SDP, with a view of converting a sparse SDP with a psd X of large dimension, into an SDP having multiple but smaller positive semidefinite matrix variables. Although, the method is not applicable to the dual slack matrix S, whose sparsity pattern is already determined, we can utilize a similar idea to generate sparse constraints, that are hopefully good.

The primal matrix X is usually dense, despite the aggregate sparsity of the C, and the A_i, $i = 1, \ldots, k$ matrices. Since

$$S \;=\; C - \sum_{i=1}^{k} y_i A_i$$

the matrix S inherits the aggregate sparsity of these matrices. Here C, $A_i, i =$

$1, \ldots, k$, and $S \in \mathcal{S}^n$. Consider a graph $G = (V, E)$ as follows. Let $V = \{1, \ldots, n\}$. Also, the edge set is given by

$$E \;=\; \{(i,j) \in V \times V : C_{ij} \;\; \text{or} A_{p_{ij}} \neq 0, p = 1, \ldots, k\}$$

Let C_r, $r = 1, \ldots, l$ be the set of maximal cliques in the graph (a clique is maximal if it is not a proper subset of any other clique in the graph).

Suppose we are given a partial symmetric matrix $\bar{X} \in \mathcal{S}^n(F, ?)$ (the notation indicates that only the entries $\bar{X_{ij}}$, $(i,j) \in F$ are known in advance). A necessary condition for $\bar{X}$ to have a positive semidefinite matrix completion is

$$\bar{X}_{C_r C_r} \;\in\; {\mathcal{S}_+}^{C_r}, \;\; r = 1, \ldots, l \tag{3.18}$$

We will refer to (3.18) as the clique-psd condition. (also see Fukuda et al [34] for more on this terminology). However, if the graph $G = (V, E)$ is chordal, i.e. every cycle in the graph of length ≥ 4 has a chord (a chord is an edge joining any two nonadjacent vertices), then we also have sufficiency, as seen from the following theorem (23) due to Grone et al [41].

Theorem 23 *Any partial symmetric matrix $\bar{X} \in \mathcal{S}^n$, satisfying the clique psd condition (3.18) can be completed to a positive semidefinite symmetric matrix X, if $G = (V, E)$ is chordal*

Assuming we work with a chordal graph, this suggests that

$$X \succeq 0 \;\; \Leftrightarrow \;\; \bar{X}_{C_r C_r} \succeq 0, \;\; r = 1, \ldots, l$$

Here X is the completion of the partial matrix $\bar{X}$, i.e. $X_{ij} = \bar{X_{ij}}$, $(i,j) \in F$.

Since the matrices $X_{C_r C_r}$ are of smaller dimension, we can easily test them for positive semidefiniteness. Also any eigenvector v corresponding to a negative eigenvalue of $X_{C_r C_r}$ can be extended to a valid cutting plane d for X, simply by assigning zeros to the components of d, not in v.

Also, if our graph is not chordal, then we then can run a minimum ordering algorithm on the aggregate sparsity matrix, and assign zero edge weights to the

additional edges appearing in *fill-in* graph. The fill-in graph can be shown to be chordal, and we proceed with the maxcut problem on this graph. It is imperative that we choose a good minimum degree algorithm to minimize the amount of fill-in. Incidentally, this approach is also discussed in Fukuda et al [34].

We utilize a similar approach for dual psd matrix S as well, although we no longer have sufficiency, i.e. we only have

$$S \succeq 0 \quad \Rightarrow S_{C_r C_r} \succeq 0, \quad r = 1, \ldots, l$$

We have dropped the bar on S. Technically, $S_{ij} = 0$, $(i,j) \notin F$. The advantages of this approach are twofold :

1. Testing the positive semidefiniteness of $S_{C_r C_r}$, $r = 1, \ldots, l$ can be done more quickly than S. In other words, the separation oracles, discussed in section 3.5.4 can be implemented more quickly.

2. We can lift any cutting plane $d \in \mathcal{R}^{|C_r|}$, to the space $\mathcal{R}^n$ by assigning zero values to all those components of d, not in C_r. This cutting plane, although slightly weaker, is sparser, and this should aid in solving the LP relaxations arising in interior point cutting plane approach.

3. Another approach we applied with moderate success is the following : We pick out the 10 largest components of the eigenvector (in magnitude) corresponding to $\lambda_{min}(S)$, and test the corresponding submatrix for positive semidefiniteness. Any eigenvector corresponding to the most negative eigenvalue is lifted to $\mathcal{R}^n$ as mentioned earlier.

3.6 The maxcut problem

We illustrate the cutting plane approach on the maxcut problem in this section. Consider the dual SDP relaxation (3.19) (Goemans and Williamson [36]) of the maxcut problem. We will present more details on this SDP relaxation, when we

discuss the maxcut problem in the next chapter.

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{s.t.} & S & = & \text{Diag}(y) - \frac{L}{4} \\ & S & \succeq & 0 \end{array} \tag{3.19}$$

where L is the Laplacian matrix of the given graph. We will work with the following relaxation (3.20)

$$\begin{array}{lll} \min & e^T y & \\ \text{s.t.} & \sum_{i=1}^{n} d_{ji}^2 y_i & \geq d_j^T \frac{L}{4} d_j, \quad j = 1, \ldots, m \end{array} \tag{3.20}$$

Our starting LP relaxation is :

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{s.t.} & y_i & \geq & \frac{L_{ii}}{4}, \quad i = 1, \ldots, n \end{array} \tag{3.21}$$

Note that, (3.21) is (3.20) with $m = n$, and $d_i = e_i$, $i = 1, \ldots, n$, and corresponds to the requirement that the diagonal entries of S (the principal minors of size 1) are nonnegative. Earlier in this chapter we mentioned that we required that our feasible set be bounded. The feasible region of (3.21) is clearly bounded only below. Thus the y can take arbitrarily large values, this is also justfied by the positive coefficients in the objective function. However, since our problem is one of minimization, we will not encounter these large y during the optimization process. Thus, the choice of a starting relaxation is dictated by the problem structure. It is not always easy to come up with a starting relaxation (see Krishnan and Mitchell [69] for more details).

The optimal solution to 3.21, is trivially $(x, y, s) = (e, \text{diag}(\frac{L}{4}), 0)$.

3.7 Computational Results

In this section, we report our preliminary computational experience. All results were obtained on a *Sun Ultra 5.6, 440 MHz* machine with *128 MB* of memory. The maxcut SDP's we consider are taken from *SDPLIB* [17], and the *DIMACS Implementation Challenge* [102].

We implemented our entire approach within the *MATLAB* environment. The LP solver was Zhang's *LIPSOL* [121]; we suitably modified the algorithm to restart the LP relaxations with a strictly feasible starting point. The SDP oracle employs the MATLAB Lanczos solver for the 2 norm, for the ∞ norm we used Vanderbei's *LOQO* [116] solver.

The parameters in our cutting plane approach are :

1. We run 200 iterations of our cutting plane scheme.

2. We add the 5 most violated cutting planes returned by the oracle in each iteration.

3. The initial tolerance $\beta = 1$, and β is reduced by 0.95 in each iteration as necessary.

All results in table 3.1 are with the ∞ norm oracle. The various columns in

Name	n	m	SDP	LDR_m (LPR_m)	UPPER	m_{lp}	EIG
mcp100 [1]	100	269	226.10	225.90(225.61)	226.58	1085	-1e-2
gpp100 [1]	100	264	221.69	221.49(221.11)	222.33	1085	-9e-3
gpp124-1 [1]	124	149	141.91	141.86(141.26)	142.52	1109	-1e-2
gpp124-2 [1]	124	318	269.64	269.35(269.04)	270.74	1109	-2e-2
gpp124-3 [1]	124	620	467.68	467.45(466.43)	468.73	1114	-2e-2
gpp124-4 [1]	124	1271	864.26	863.19(861.90)	867.43	1119	-4e-2
gpp250-1 [1]	250	331	317.23	316.31(315.13)	319.24	1245	-2e-2
gpp250-2 [1]	250	612	531.78	529.86(528.18)	535.26	1245	-9e-2
gpp250-3 [1]	250	1283	980.91	978.15(976.22)	986.36	1245	-5e-2
gpp250-4 [1]	250	2421	1681.96	1678.76(1674.26)	1689.59	1250	-6e-2
gpp500-1 [1]	500	625	598.11	597.66(594.18)	601.77	1995	-6e-2
gpp500-2 [1]	500	1223	1070.00	1069.03(1059.96)	1075.02	1995	-3e-2
gpp500-3 [1]	500	2355	1846.60	1844.39(1837.71)	1855.84	2000	-3e-2
gpp500-4 [1]	500	5120	3566.74	3560.82(3543.00)	3585.23	1575	-5e-2
maxG11 [1]	800	1600	629.16	627.95(617.75)	641.87	2300	-3e-2
maxG32 [1]	2000	2000	1567.64	1547.06(1531.87)	-	3000	-9e-2
maxG51 [1]	1000	5909	4003.81	3990.68(3985.91)	-	2000	-7e-2
toruspm-8-50 [2]	512	1536	527.81	525.64(522.43)	532.87	2007	-1e-2
torusg3-8 [2]	512	1536	457.36	457.39(445.50)	462.86	2012	-3e-2

Table 3.1: Cutting plane maxcut LP relaxations

the table (3.1) represent the following :

1. **Name :** The max cut instance.
2. **n :** The number of nodes in the graph.
3. **m :** The number of edges in the graph.
4. LDR_m (LPR_m) **:** The objective values of the dual and primal LP relaxations. Note that only the primal relaxation is a guaranteed lower bound.
5. **UPPER :** The upper bound returned by the cutting plane approach.
6. m_{lp} **:** Number of constraints in the LP relaxation.
7. **EIG :** The value of $\lambda_{min}(S)$ at termination.

For the results mentioned in table (3.1), we do not drop any constraints. The smaller instances, i.e. the graphs with $100 - 250$ nodes were solved under an hour. The sparser graphs with 500 nodes (gpp500-1 and gpp500-2) took about an hour, and we needed about 3 hours for the remaining problems in the set. The sparsity of the Laplacian matrix $\frac{L}{4}$ (determined by the sparsity of the graph) does affect the speed of the oracle. The interior point methods solve all these problems in under an hour.

We are still experimenting with designing effective techniques to dropping constraints that no longer appear important. This will give a better estimate of the times involved. More details will be available in the forthcoming report Krishnan and Mitchell [71].

Finally, we compare the 2 norm and the ∞ norm separation oracles (for an equivalent number of iterations). We also compare both approaches with the bundle LP approach discussed in Chapter 2. The results are summarized in table (3.2). We ran the cutting plane approach for 100 iterations in all. All tests were performed in under an hour.

The various columns in table (3.2) represent :

[1]SDPLIB
[2]DIMACS
[1]60 iterations only

Name	SDP	LP_B	m_B	CP_2	m_2	CP_∞	m_∞
toruspm-8-50	527.81	525.91	528	-	-	520.52	986
torusg3-8	457.36	454.00	525	-	-	450.73	982
mcp100	226.16	225.75	110	219.85	384	220.78	515
mcp124-1	141.99	141.06	144	140.89	427	141.46	563
mcp124-2	269.88	268.97	135	266.11	410	268.17	532
mcp124-3	467.75	467.37	135	459.61	381	464.73	500
mcp124-4	864.41	863.72	134	849.91	348	859.25	516
mcp250-1	317.26	317.18	260	305.86	562	315.22	679
mcp250-2	531.93	531.18	264	511.45	565	527.03	693
mcp250-3	981.17	980.32	263	953.40	565	972.03	651
mcp250-4	1681.96	1679.70	263	1650.87	573	1663.27	640
mcp500-1	598.15	594.12	510	547.41	825	593.05	939
mcp500-2	1070.06	1069.90	513	989.11	814	1057.63	921
mcp500-3	1847.97	1843.20	514	1777.14	882	1825.03	901
mcp500-4 [1]	3566.74	3559.80	515	3479.80	875	3524.73	799

Table 3.2: Comparison of the various LP relaxations

1. **Name :** Problem Name.
2. **SDP :** The SDP objective value.
3. LP_B **:** The bundle LP objective value.
4. m_B **:** Number of constraints in the bundle LP relaxation.
5. CP_2 **:** The objective value of the cutting plane approach that uses the 2 norm oracle.
6. m_2 **:** The number of constraints in the cutting plane relaxation using the 2 norm.
7. CP_∞ **:** The objective value of cutting plane approach that uses the ∞ norm oracle.
8. m_∞ **:** The number of constraints in the cutting plane relaxation that uses the ∞ norm.

3.8 Conclusions

Here are some conclusions based on our preliminary computational experiments.

1. There is tremendous flexibility available in these interior point cutting plane approaches. There are a number of adjustable parameters such as the tolerance to which one solves the LP relaxations, the number of cutting planes added in each iteration, the technique for restarting the LP relaxations, and most importantly the choice of a problem specific oracle. For instance in the max-cut problem, the ∞ norm oracle performs well, since it mimics the incidence vectors of cuts.

2. The bundle LP approach discussed in Chapter 2 is superior to the cutting plane approaches we discussed in this chapter. This is presumably because the bundle approach better exploits the geometry of the SDP problem.

3. The cutting plane LP approaches for SDP (and other convex programming problems) are quite different from the traditional cutting plane approaches for solving integer programming problems. In the latter case we can utilize the facets of the underlying integer polytope, to generate the best cutting planes available. On the other hand in our case, the oracle returns different cutting planes, some of which are facets, and others that are lower dimensional faces of the SDP cone. This is intimately related to the multiplicity of $\lambda_{min}(S)$. Since we are trying to force this eigenvalue to zero, we are implicitly altering its multiplicity during the course of the cutting plane scheme. For instance in 2 norm case, the cutting plane approach should slow down towards optimality, where the multiplicity of $\lambda_{min}(S)$ is high since the smaller eigenvalues coalesce together.

4. The most time consuming step in each iteration is the one taken by the oracle. It is imperative that we keep the size of this oracle subproblem small. In this regard, we discussed some strategies in section 3.5.5.

5. We need to improve our technique of generating lower bounds. We presented

some strategies in section 3.5.1 where we exploit the information available in the eigenvector corresponding to $\lambda_{min}(S)$; however more work needs to be done in this matter. Equally important is choosing a good criteria for dropping constraints. The oracles in the SDP approach return dense constraints; moreover we solve these relaxations using an interior point scheme. Thus it is important to keep the size of the LP relaxations small.

6. The aim is utilize this approach in a branch and cut approach for combinatorial optimization problems. We discuss such an approach on the maxcut problem in the next chapter. This gives us a way to combine the nonpolyhedral cuts returned by the interior point cutting plane approach, with the facet defining inequalities in an overall LP approach for integer programming problems.

7. A future aim is to extend this approach to other SDP's with bounded feasible set such as min bisection, k way equipartition, Lovasz theta, and box constrained quadratic programs. This is not entirely straightforward, since one needs to get hold of a good initial LP relaxation of the SDP problem at hand.

3.9 Acknowledgments

The author would like to thank Beresford Parlett for illuminating discussions regarding improving the upper bound in the cutting plane approach. More details can be found in Parlett and Krishnan [98].

CHAPTER 4
Cutting plane approaches for the maxcut problem

4.1 Abstract

We investigate solution of the maximum cut problem using a cutting plane semidefinite programming (SDP) approach. The dual of the well-known SDP relaxation of maxcut is formulated as a semi-infinite linear programming problem, which is solved within a cutting plane algorithm. Cutting planes based on the polyhedral theory of the maxcut problem are then added to the primal problem in order to improve the SDP relaxation. We provide computational results, and compare these results with a standard SDP cutting plane scheme.

4.2 The Maximum Cut Problem

Let $G = (V, E)$ denote an edge weighted undirected graph without loops or multiple edges. Let $V = \{1, \ldots, n\}$, $E \subset \{\{i, j\} : 1 \leq i < j \leq n\}$, and $w \in \mathcal{R}^{|E|}$, with $\{i, j\}$ the edge with endpoints i and j, and weight w_{ij}. We assume that $n = |V|$, and $m = |E|$. For $S \subseteq V$, the set of edges $\{i, j\} \in E$ with one endpoint in S and the other in $V \backslash S$ form the *cut* denoted by $\delta(S)$. We define the weight of the cut as $w(\delta(S)) = \sum_{\{i,j\} \in \delta(S)} w_{ij}$. The *maximum cut problem*, denoted as (MC), is the problem of finding a cut for which the total weight is maximum. (MC) can be formulated as

$$\max\{w(\delta(S)) | S \subseteq V\} \qquad (MC) \tag{4.1}$$

We illustrate the maxcut problem in figure (4.1). The problem finds numerous applications in VLSI and circuit board design, network design, and finally Ising spin glass problems in statistical physics. One must mention that the Ising spin glass problems are sources of the maxcut problem where the edge weights can be negative. The maximum cut problem is one of the original NP complete problems (Karp [63]).

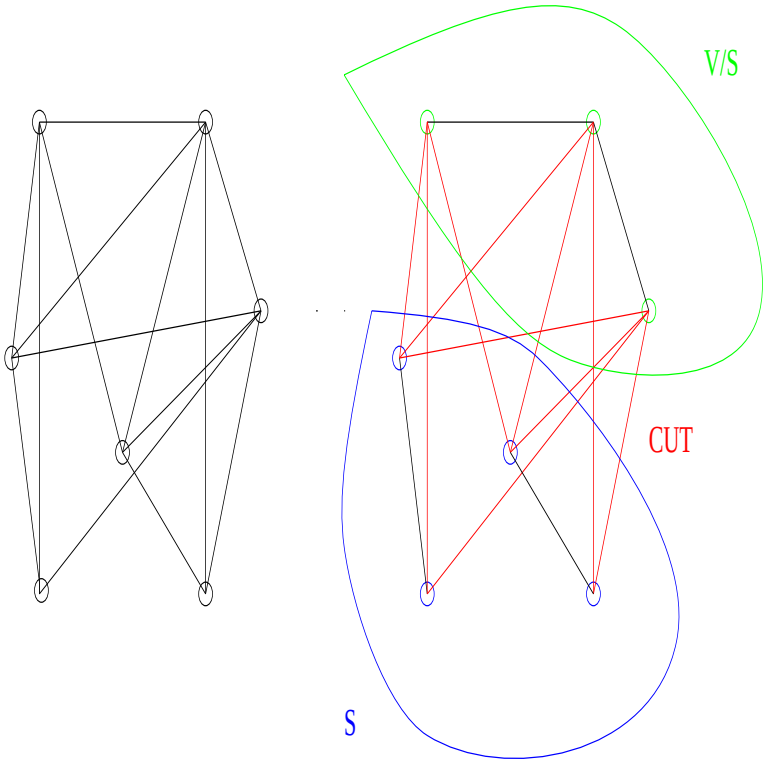


Figure 4.1: A graph and its maxcut

However there are several classes of graphs for which the maximum cut problem can be solved in polynomial time; for instance planar graphs, graphs with no K_5 minor etc. A good survey on the maxcut problem appears in Poljak and Tuza [103]. We discuss various approaches to solving the maxcut problem in this chapter. The aim is to try and solve the maxcut problem down to optimality, using convex tractable relaxations of the problem, cutting planes, heuristics, and branch and bound.

A quick word on notation : we represent vectors and matrices, by lower and upper case letters respectively. For instance d_{ij} denotes the jth entry of the vector d_i in a collection, whereas X_{ij} denotes the entry in the ith row and jth column of matrix X, we also use MATLAB like notation frequently in this chapter. For instance $y(n+1:m)$ will denote components $n+1$ to m of vector y, while $A(n+1:m, 1:p)$ will denote the sub-matrix obtained by considering rows $n+1$ to m, and the first p columns of matrix A. Finally $d_j.^2$ refers to a vector obtained by squaring all the components of d_j. A glossary of all the notation in this chapter can be found in Appendix A.

The aim of this chapter is to provide a self contained introduction to the maxcut problem; we also provide references along the way. The chapter is organized in a manner that we have all the details at hand to describe our cutting plane approach, which appears in section 4.6. This chapter is organized as follows : section 4.3 deals with a linear programming LP formulation of the maxcut problem, together with an interior point LP cutting plane approach to solving the maxcut problem, section 4.4 deals with semidefinite formulations of the maxcut problem, the Goemans and Williamson approximation algorithm for the maxcut problem, and a prototype SDP cutting plane approach for the maxcut problem. Section 4.5 describes other approaches for the maxcut problem, in section 4.6 we present our SDP cutting plane approach for the maxcut problem. We present some computational results in section 4.7, and our conclusions in section 4.8.

4.3 Linear programming formulations of the maxcut problem

Consider a variable x_{ij} for each edge $\{i,j\}$ in the graph. Let x_{ij} assume the value 1 if edge $\{i,j\}$ is in the cut, and 0 otherwise. The maxcut problem can then be modeled as the following integer programming problem.

$$\begin{array}{lll} \max & \sum_{i=1}^{n} \sum_{\{i,j\} \in E} \frac{1}{2} w_{ij} x_{ij} & \\ \text{subject to} & x & \text{is the incidence vector a cut} \end{array} \tag{4.2}$$

Here n is the number of vertices in the graph, and the factor $\frac{1}{2}$ appears since each edge in the graph is considered twice.

Let CUT(G) denote the convex hull of the incidence vectors of cuts. Since maximizing a linear function over a set of points is equivalent to maximizing it over the convex hull of these set of points, we can also write (4.2) as the following linear program.

$$\begin{array}{ll} \max & c^T x \\ \text{s.t.} & x \in \mathrm{CUT}(G) \end{array} \tag{4.3}$$

Here c and $x \in \mathcal{R}^m$, where m is the number of edges in the graph. We usually index c and x with the subscript e to relate them to the edges in the graph. However from time to time we drop the subscript e as appropriate. If we had an exact description of the convex hull $\mathrm{CUT}(G)$, or more importantly had an oracle capable of solving the separation problem with respect to this polytope in polynomial time, then we could use this oracle in conjunction with the ellipsoid algorithm to solve the maxcut problem in polynomial time, thereby showing that $P = NP$. This follows from the equivalence of separation and optimization which appears in Grotschel et al [42]. Unfortunately an exact description is unknown, nor do we always have polynomial time separation oracles for some of the inequalities that describe the maxcut polytope, owing to the *NP complete* nature of the problem.

It is instructive to study facets (faces of the maxcut polytope), since these provide good separation inequalities in a cutting plane approach. Numerous facets of the maxcut polytope have been studied, but the ones widely investigated in connection with a cutting plane approach are the following inequalities

$$\begin{array}{rcll} 0 \leq & x_e & \leq 1 & \forall e \in E, \\ x(F) - x(C \backslash F) & \leq & |F| - 1 & \forall \text{ circuits } C \subseteq E \\ & & & \text{and all } F \subseteq C \text{ with } |F| \text{ odd} \end{array} \tag{4.4}$$

Here $x(S) = \sum_{\{i,j\} \in S} x_{ij}$ for any $S \subseteq E$. It is easy to see that these are valid inequalities for the maxcut polytope. The first set of inequalities define facets of $\mathrm{CUT}(G)$, if e is not contained in a triangle. The second set define facets for each $F \subseteq C$, with $|F|$ odd, if C has no chord. We will call a graph *chordal* if any cycle C of length ≥ 4 has a chord. These chordal graphs will play a special role as shall be seen in section (4.6.1). The latter inequalities can be derived from the observation that every cycle and cut intersect in an even number of edges. Moreover, the integral vectors satisfying (4.4) are vertices of $\mathrm{CUT}(G)$. So we indeed have an

integer programming formulation of the maximum cut problem, namely

$$\begin{array}{lrcll} \max & & c^T x & & \\ \text{s.t.} & x(F) - x(C\backslash F) & \leq & |F| - 1 & \forall \text{ circuits } C \subseteq E \\ & & & & \text{and all } F \subseteq C \text{ with } |F| \text{ odd} \\ & x & \geq & 0 & \\ & x & \leq & e & \\ & x & \in & \{0,1\}^m & \end{array} \tag{4.5}$$

Incidentally, these inequalities define a polytope known as the *odd cycle polytope*. Barahona and Mahjoub [10] show that we can drop the integrality restriction in (4.5), and solve the maxcut problem simply as an LP, if G does not have any subgraph contractible to K_5. This includes among others planar graphs, so we can indeed solve these maxcut instances in polynomial time.

Although there are an exponential number of linear constraints in (4.5), Barahona and Mahjoub [10] describe a polynomial time separation oracle for the inequalities (4.4) that involves solving n shortest path problems on an auxiliary graph with twice the number of nodes, and four times the number of edges. Thus it is possible to optimize a linear function over the odd cycle polytope in polynomial time.

The Barahona-Mahjoub Separation oracle

1. Construct a graph $H = (V' \cup V'', E' \cup E'' \cup E''')$, consisting of two disjoint copies $G' = (V', E')$ and $G'' = (V'', E'')$ of G, and an additional set of edges E''' that contains for every $\{i, j\} \in E$, the edges $\{i', j''\}$, and $\{i'', j'\}$. The edges $\{i', j'\} \in E'$, and $\{i'', j''\} \in E''$ have the weight x_{ij}, while the edges $\{i', j''\}$ and $\{i'', j'\} \in E'''$ get the weight $(1 - x_{ij})$.

2. For each node $i \in V$, we calculate a shortest (with respect to the weights just defined) path in H from $i' \in V'$ to $i'' \in V''$. Such a path contains an odd number of edges of E''' and corresponds to a closed walk in G containing vertex i.

3. If the shortest of these $[i', i'']$, $i = 1, \ldots, n$ has length at least 1, then x_{ij} satisfies all the odd cycle inequalities. This is obtained by rewriting the odd cycle inequality (4.4) as

$$\sum_{ij \in F}(1 - x_{ij}) + \sum_{ij \in C \setminus F} x_{ij} \quad \geq \quad 1$$

Any shortest path $[i', i'']$ with a length less than 1 gives a violated odd cycle inequality.

4. Our aim is to add the odd cycle inequalities that are violated as much as possible. If we define the violation of an ith odd cycle inequality as $\text{VIO}_i = (1 - \text{length}[i', i''])/|C|$, then we want to add cuts, where this violation is as large as possible. The length of the shortest path is the sum of the weight of the edges in the shortest path, and $|C|$ is the length of the cycle.

Since a shortest path algorithm can be solved in $O(n^2)$ time for a complete graph, the entire separation routine can be carried out in $O(n^3)$ arithmetic operations.

Another idea that has been used to construct cutting planes for $\{0, 1\}$ integer programming problems like the maxcut problem, is an implicit way, which consists of trying to represent the cut polytope CUT as the projection of another polytope Q (say) lying in a higher dimensional space. The rationale is that the projection of Q may have a more complicated facial structure than Q itself. Several schemes have been proposed for constructing projection representations for the CUT polytope; see in particular Balas et al [8], Lovasz and Schrijver [79], Sherali and Adams [108], and Lasserre [72]. A common feature of these methods is that they construct a hierarchy

$$K \supseteq K_1 \supseteq \ldots \supseteq K_d \supseteq CUT$$

of relaxations of CUT obtained as projections of higher dimensional polyhedra Q_i, $i = 1, \ldots, d$, that finds CUT in d steps, i.e. $K_d = CUT$. Also,

$$d \quad \leq \quad \# \text{ integer variables in the formulation}$$

The number of constraints after d liftings is 2^d (Number of original constraints). An important point is that if we can optimize a linear function over K in polynomial time, then the same holds for K_t for any *fixed* t. The relaxations are linear in Sherali-Adams, in the case of Lovasz-Schrijver and Lasserre semidefinite.

We must mention that there are two equivalent formulations of the maxcut problem, the $\{0,1\}$ formulation considered earlier, and the $\{-1,1\}$ formulation, which will be useful when we consider semidefinite formulations of the maxcut problem in the next section. To provide an equivalence we consider an affine transformation $y = \frac{e}{2} + \frac{x}{2}$ of an vector $x \in \{-1,1\}^n$ to a 0-1 vector $y \in \{0,1\}^n$.

The Barahona-Mahjoub separation oracle can be utilized in a cutting plane LP approach for the maxcut problem. In this section, we describe an interior point cutting plane LP approach to the maxcut problem due to Mitchell [82]. We only highlight the salient features; the idea is to illustrate the numerous algorithmic decisions that have to made in any cutting plane scheme.

An LP cutting plane approach for maxcut

1. **Initialize** : The initial relaxation is

$$\begin{array}{lll} \max & & c^T x \\ \text{s.t.} & 0 \leq x \leq & e \end{array} \tag{4.6}$$

   where $c, x \in \mathcal{R}^m$, where m is the number of edges in the graph. It is easy to see that an optimal solution is given by

$$x = \begin{cases} 1 & : \text{ if } c_i > 0 \\ 0 & : \text{ otherwise} \end{cases}$$

2. **Solve the current LP relaxation** approximately using a primal-dual interior point scheme. The LP relaxation provides an upper bound on the optimal value of the maxcut problem.

3. **Solve the separation problem** using heuristics, and calling the Barahona-Mahjoub separation oracle if necessary, to find any odd cycle inequalities vi-

olated by the current solution. Bucket sort the constraints by violation, and add the most violated ones to the current LP relaxation.

4. **Use a heuristic** to generate good incidence cut vectors. These include rounding the current LP solution, or using more sophisticated local search procedures like Kernighan-Lin to be described later. The heuristic provides a lower bound on the optimal maxcut value.

5. **Check for termination** If the difference between the upper and lower bounds is small, stop with optimality. Else continue with steps 2-4. If no violated odd cycle inequalities were found in the current iteration resort to branch and bound to complete the solution.

A number of important features have been omitted, and these can be found in Mitchell [82]. Mitchell [82] reports solving large scale ± 1 Ising spin glass problems, with integer optimal solutions fairly quickly using this approach. Barahona and Anbil [9] are able to solve large instances using a volume algorithm, where an interior point scheme is infeasible due to high storage requirements. A few points are now in order :

1. The cutting plane LP approach described above uses an interior point scheme rather a simplex scheme to solve the LP relaxations, since this results in better cutting planes. As remarked in Mitchell [82] it suffices to solve the LP relaxations *approximately* gradually tightening the degree of accuracy, a feature available only in an interior point approach. Thus the success of the cutting plane approach also depends on the LP solver employed. In this regard we mention the volume algorithm due to Barahona and Anbil [9], a subgradient algorithm that can be used to generate approximate solutions to the LP relaxations quickly.

2. The idea of using an interior point scheme presents its own share of difficulties. In particular we need a warm start strategy to restart the LP relaxations, especially the primal where we add cutting planes. Since the cuts returned by the oracle are *deep*, the previous primal solution is no longer strictly feasible.

This restart is unlike the simplex method, where we can re-optimize using the dual simplex procedure. However we can use the special structure of the maxcut problem here : Mitchell [82] always restarts the primal by backtracking from the current LP solution towards the point $x = \frac{e}{2}$, which is strictly within the maxcut polytope.

3. It is important to select a *good* subset of violated inequalities, if too many are found. Also, it is important to drop inequalities that are not important, to keep the LP relaxations small.

4. It is important to consider which cutting planes to use; and also use cheap heuristics to find such cutting planes quickly. A good idea is to test all the triangle inequalities by complete enumeration, and when we run out of these call the Barahona-Mahjoub separation oracle.

5. We also need good heuristics that generate lower bounds. Mitchell [82] uses a randomized rounding procedure to generate good incidence cut vectors, and then improves it using a local search heuristic like Kernighan and Lin. The local search procedure uses a breadth first search procedure to explore all paths from a vertex, and all the vertices in a path are swapped to the other side, if this results in an improvement. Also, see Berry and Goldberg [15] for similar heuristics.

6. Since most combinatorial optimization problems are *NP hard*, we will need to resort to branching at some stage. Without branching we are optimizing over the odd cycle polytope, which we have seen contains the maxcut polytope. Again it is important to consider warm start strategies within the branch and bound context.

7. Last but not the least, it is also important to consider a good initial relaxation of the combinatorial optimization problem at hand. We will see in the next section that semidefinite relaxations of the maxcut problem resolve this issue, since they come with performance guarantees.

4.4 Semidefinite formulations of the maxcut problem

In this section we consider semidefinite programming (SDP) formulations of the maxcut problem. In particular we consider Goemans and Williamson's celebrated 0.878 approximation algorithm for the maxcut problem, based on this SDP formulation. We will later consider a semidefinite cutting plane approach for the maxcut problem.

As mentioned before the maxcut problem can be written as

$$\max_{S\subseteq V} \quad \sum_{\{i,j\}\in\delta(S)} w_{ij} \qquad (MC) \tag{4.7}$$

Without loss of generality, we can assume that our graph is complete. In order to model an arbitrary graph in this manner, define $w_{ij} = 0$, $\{i,j\} \notin E$. $A = (w_{ij}) \in \mathcal{S}^n$ is referred to as the weighted adjacency matrix of the graph.

In order to model this as an integer program consider cut vectors $x \in \{-1,1\}^n$ with $x_i = 1$ if $i \in S$, and $x_i = -1$ for $i \in V\ S$. Consider the following problem

$$\max_{x\in\{-1,1\}^n} \quad \sum_{i,j=1}^n w_{ij}\frac{1-x_ix_j}{4} \tag{4.8}$$

A factor of $\frac{1}{2}$ accounts for the fact that each edge is considered twice. Moreover the expression $\frac{(1-x_ix_j)}{2}$ is 0 if $x_i = x_j$, i.e. if i and j are in the same set, and 1 if $x_i = -x_j$. Thus $\frac{(1-x_ix_j)}{2}$ yields the *incidence vector* of a cut associated with a cut vector x, evaluating to 1 if and only if edge $\{i,j\}$ is in the cut. Exploiting the fact that $x_i^2 = 1$, we have

$$\begin{aligned} \tfrac{1}{4}\sum_{i,j=1}^n w_{ij}(1-x_ix_j) &= \tfrac{1}{4}\sum_{i=1}^n(\sum_{j=1}^n w_{ij}x_i^2 - \sum_{j=1}^n w_{ij}x_ix_j) \\ &= \tfrac{1}{4}x^T(\mathrm{Diag}(Ae) - A)x \end{aligned} \tag{4.9}$$

The matrix $L = \mathrm{Diag}(Ae) - A$ is called the *Laplacian* matrix of the graph G. Letting $C = \frac{L}{4}$, we find that the maxcut problem can be interpreted as a special case of the following more general $\{+1,-1\}$ integer programming problem.

$$\max_{x\in\{-1,1\}^n} \quad x^T C x \tag{4.10}$$

We are now ready to derive a semidefinite programming relaxation for the maxcut problem. First note that $x^T C x = \text{Trace}(C x x^T)$. Since $x \in \{-1,1\}^n$, the matrix xx^T is positive semidefinite, its diagonal entries are equal to one. Also it is a rank one matrix. Thus (4.10) is equivalent to the following problem.

$$\begin{array}{lrcl} \max & \frac{1}{4}L \bullet X & & \\ \text{s.t.} & \text{diag}(X) & = & e \\ & X & \succeq & 0 \\ & \text{rank}(X) & = & 1 \end{array} \tag{4.11}$$

This problem is not yet a semidefinite program, owing to the rank one constraint. We demonstrate the non-convex nature of this problem, by noting the equivalence of (4.11) to the following problem (Goemans and Williamson [36] and Nesterov [89]).

$$\begin{array}{lrcl} \max & \frac{2}{\pi}C \bullet \arcsin(X) & & \\ \text{s.t.} & \text{diag}(X) & = & e \\ & X & \succeq & 0 \end{array} \tag{4.12}$$

The objective function in (4.12) is non-convex. Interestingly we can also write (4.11) as the following semidefinite programming problem, with ± 1 integrality restrictions.

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \text{diag}(X) & = & e \\ & X & \succeq & 0 \\ & X_{ij} & \in & \{-1,1\} \;\; \forall i < j \end{array} \tag{4.13}$$

This follows Theorem 24 (Laurent and Poljak [74])

Theorem 24 *Let $X \in \{X \succeq 0 : diag(X) = e\}$. Then if $X_{ij} \in \{-1,1\}$, $\forall i \neq j$,*

then $X = xx^T$, *with* $x \in \{-1, 1\}^n$.

Dropping the rank one restriction, we arrive at the following semidefinite programming relaxation of the maxcut problem

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \text{diag}(X) & = & e \\ & X & \succeq & 0 \end{array} \tag{4.14}$$

and its dual

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{s.t.} & S & = & \text{Diag}(y) - \frac{L}{4} \\ & S & \succeq & 0 \end{array} \tag{4.15}$$

Oustry and Lemarechal [94] arrive at the SDP relaxation (4.14) by taking the dual of the Lagrangian dual of (4.8), which incidentally is (4.15). We will refer to the feasible region of (4.14) as the *elliptope*. In other words the elliptope is the set of symmetric positive semidefinite matrices, whose diagonal entries are all one. A point that must be emphasized is that the elliptope is no longer a polytope. Thus (4.14) is actually a non-polyhedral relaxation of the maxcut problem. The feasible region for a graph with three nodes is shown in figure (4.2); a convex region obtained by the intersection of the affine subspace $\text{diag}(X) = e$, with the cone $X \succeq 0$. The plot was obtained by setting $X = \begin{bmatrix} 1 & x & y \\ x & 1 & z \\ y & z & 1 \end{bmatrix}$, and plotting $\det(X) = 0$, for $-1 \leq \begin{bmatrix} x \\ y \\ z \end{bmatrix} \leq 1$. The maxcut polytope is the tetrahedron obtained by the convex combination of the four points $[1, -1, -1], [-1, 1, -1], [-1, -1, 1], [1, 1, 1]$. There is not much to choose between the elliptope and the maxcut polytope here, and the former contains the latter.

These semidefinite programs satisfy strong duality, since $X = I$ is strictly feasible in the primal, and we can generate a strictly feasible dual solution by assigning

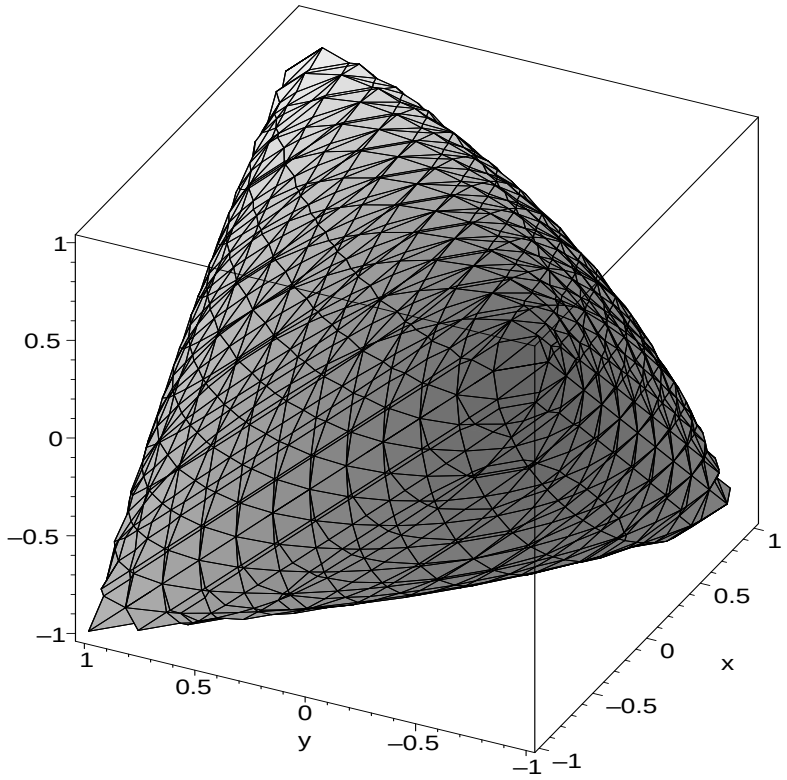


Figure 4.2: The feasible set of (4.14) for $n = 3$

y an arbitrary positive value. In fact setting $y_i = 1 + \sum_{j=1}^{n} |\frac{L_{ij}}{4}|$ and $S = \text{Diag}(y) - \frac{L}{4}$ should suffice.

In a semidefinite cutting plane scheme for the maxcut problem, we would begin with (4.14) as our starting SDP relaxation. It is interesting to relate (4.14) with (4.6) for the LP cutting plane approach. The constraints $\text{diag}(X) = e$, and $X \succeq 0$ together imply that $-1 \leq X_{ij} \leq 1$. We now transform the problem into $\{0, 1\}$ variables using the transformation $y_{ij} = \frac{1-X_{ij}}{2}$. Thus we have $\frac{L}{4} \bullet X = \frac{1}{2} \sum_{i=1}^{n} \sum_{\{i,j\} \in E} w_{ij} y_{ij}$. Also we have $0 \leq y_{ij} \leq 1$, $\forall \{i, j\} \in E$. Thus it is easy to see that (4.14) is actually tighter than (4.6). Actually, we can also establish a performance guarantee as shall be seen later.

We now present the Goemans and Williamson approximation algorithm for the maxcut problem. Since we shall be using this algorithm within our cutting plane approach, we will outline the steps clearly below.

The Goemans-Williamson (GW) approximation algorithm for maxcut

1. Solve the SDP relaxation (4.14) to get a primal matrix X.

2. Compute V such that $X = V^T V$. This can be done either by computing the Cholesky factorization of X, or by computing the spectral decomposition of $X = P\Lambda P^T$, and then computing $V = \sqrt{\Lambda} P^T$.

3. Randomly partition the unit sphere in $\mathcal{R}^n$ into two half spheres H_1 and H_2 (the boundary in between can be on either side), and form the bipartition consisting of $V_1 = \{i : v_i \in H_1\}$ and $V_2 = \{i : v_i \in H_2\}$. The partitioning is carried out in practice by generating a random vector r on the unit sphere, and assigning i to V_1 if $v_i^T r \geq 0$, and V_2 otherwise. In fact we repeat the procedure a number of times, and pick the best cut obtained.

Hereafter, we refer to step 3 as the *GW rounding procedure*. It is important to note that step 3 gives a lower bound on the optimal maxcut solution, while the SDP relaxation in step 1 gives an upper bound.

A few notes on the GW rounding procedure : For any factorization of $X = V^T V$ in step 2, the columns of V yield vectors v_i, $i = 1, \ldots, n$. Since we have $\text{diag}(X) = e$, each vector is of unit length, i.e. $||v_i|| = 1$. Associating a vector v_i with node i, we may interpret v_i as the relaxation of $x_i \in \{-1, 1\}$ to the n dimensional unit sphere. Thus we are essentially solving

$$\begin{array}{lll} \max & \sum_{i,j=1}^{n} \frac{L_{ij}}{4} v_i^T v_j & \\ \text{s.t.} & ||v_i|| = 1 & \forall i = 1, \ldots, n \\ & v_i \in \mathcal{R}^n & \end{array} \tag{4.16}$$

This vector formulation provides a way to interpret the solution to the maxcut SDP. Since v_i and v_j are unit vectors, $v_i^T v_j$ is the cosine of the angle between these vectors. If all the edge weights w_{ij} are nonnegative, the off diagonal entries of the Laplacian matrix are negative. Thus if the angle between the vectors is large, we should separate the corresponding vertices, if it is small we put them in the same set (since this would improve the objective function in the vector formulation). In order to avoid conflicts, Goemans and Williamson [36] consider the random hyperplane

technique mentioned in step 3. Step 3 is in accord with our earlier intuition, since vectors with a large angle between them are more likely to be separated, since the hyperplane can end up between them.

Finally the celebrated result due to Goemans and Williamson.

Theorem 25 *Let us assume that all the edges in the graph have nonnegative edge weights. Also let $E(cut)$ denote the expected value of the cut generated by the rounding procedure. We then have*

$$\begin{aligned} \textit{Optimal maxcut solution} &\geq E(cut) \\ &\geq 0.878 \ \textit{Maxcut SDP solution} \end{aligned}$$

Since the SDP provides an upper bound on the maxcut solution, Theorem 25 provides an 0.878 approximation algorithm for the maxcut problem, when all the edge weights are nonnegative. On the negative side Hastad [44] showed that it is NP-hard to approximate the maxcut problem to within a factor of 0.9412.

If we include negative edge weights, then Nesterov [89] showed that the GW rounding procedure gives an $\frac{2}{\pi}$ approximation algorithm for the maxcut problem.

We can improve the GW SDP relaxation using the following linear inequalities.

1. **The chord-less odd cycle inequalities** :
   In the $\{-1, 1\}$ setting the odd cycle inequalities (4.4) are

$$\begin{aligned} X(C\backslash F) - X(F) &\leq |C| - 2 \\ &\text{for each cycle } C, F \subset C, |F| \text{ odd} \end{aligned} \tag{4.17}$$

   These include among others the triangle inequalities. Since $\text{diag}(X) = e$, and $X \succeq 0$ imply $-1 \leq X_{ij} \leq 1$, the feasible region of the initial GW SDP relaxation (4.14) intersected with the odd cycle inequalities (4.17) is contained within the odd cycle polytope. We should thus get tighter upper bounds on the maxcut value.

2. **The hypermetric inequalities** :

These are inequalities of the form (4.18)

$$bb^T \bullet X \geq 1 \qquad \text{where } b \in \mathcal{Z}^n, \quad \sum_{i=1}^n b_i \text{ odd} \qquad \text{and } \min\{(b^T x)^2 : x \in \{-1,1\}^n\} = 1. \tag{4.18}$$

For instance, the triangle inequality $X_{ij} + X_{ik} + X_{jk} \geq -1$ can be written as a hypermetric inequality by letting b to be the incidence vector of the triangle (i,j,k). On the other hand the other inequality $X_{ij} - X_{ik} - X_{jk} \geq -1$ can be written in a similar way, except that $b_k = -1$. Although there are a countably infinite number of them, these inequalities also form a polytope known as the *hypermetric polytope*. Helmberg and Rendl [53] describe simple heuristics to detect violated hypermetric inequalities.

3. **Lift and project cutting planes** :
   Recently Iyengar and Cezik [60] developed a lift and project cutting plane method, where they extend the ideas of Balas et al [8] for the LP (discussed previously) to semidefinite programs with $\{-1,1\}$ integrality restrictions. From (4.13) it is clear that the maxcut problem falls in this category. The essential idea is to tighten the current relaxation by imposing an $\{-1,1\}$ restriction on one of the variables. A deep cutting plane is obtained by solving an SDP. However solving this SDP to find the most violated inequality is prohibitively expensive.

Interestingly although the additional inequalities improve the SDP relaxation, they do not necessarily give rise to better approximation algorithms. On the negative side Karloff [61] exhibited a set of graphs for which the optimal solution to (4.14) satisfies all the triangle inequalities as well, so after the GW rounding procedure we are still left with a 0.878 approximation algorithm.

Goemans and Williamson [36] show that the randomized rounding procedure performs well if the ratio of the weight of the edges in the cut, to those in the graph is more than 85%. If this is not true, then it pays to introduce more randomness in the rounding procedure. Zwick [123] suggests that we consider $(\gamma I + (1-\gamma)X)$ rather than X, for some appropriate $\gamma \in [0,1]$.

More recently Anjos and Wolkowicz [7] presented a strengthened semidefinite relaxation of the maxcut problem. Their SDP relaxation is tighter than the GW SDP relaxation together with all the triangle inequalities. To arrive at their relaxation they add certain redundant constraints from the maxcut problem to (4.14), and consider the Lagrangian dual of the Lagrangian dual to this problem. On the negative side though this strengthened SDP relaxation is fairly expensive to solve.

The solution obtained by the randomized rounding procedure can be further improved using a Kernighan-Lin local search heuristic. Since this heuristic plays an important role in an SDP cutting plane approach, we present the details (also see Papadimitriou and Steiglitz [96]).

The Kernighan and Lin (KL) heuristic

1. For each vertex i, $i = 1, \ldots, n$, compute the external and internal costs $E(i)$ and $I(i)$. Let us refer to the two sets in the bipartition as S and T respectively. If for instance a vertex a is in S, then

$$\begin{aligned} E(a) &= \textstyle\sum_{i \in T} w_{ai} \\ I(a) &= \textstyle\sum_{i \in S} w_{ai} \end{aligned}$$

2. For each vertex i, compute $D^1(i) = E(i) - I(i)$, $i = 1, \ldots, n$. For instance sending the vertex a above from S to T will result in a reduction of $D(a)$. The objective value of the cut *drops* by an equivalent amount.

3. In the k iteration, choose a vertex i, $i = 1, \ldots, n$ such that $D^k(i)$ is a minimum (not necessarily negative; a negative value will definitely improve the value of the new cut). Let us call the vertex b. Let us assume for simplicity that b goes from S to T. Let $g(k) = D^k(b)$.

4. Now mark b as *tabu*. The vertex b will not be considered in future interchanges. Update $D(v)$ for all vertices v adjacent to b as follows :

$$\begin{aligned} D^{k+1}(v) &= D^k(v) + 2w_{bv}, \quad \forall v \in S \\ D^{k+1}(v) &= D^k(v) - 2w_{bv}, \quad \forall v \in T \end{aligned}$$

Also, $D^{k+1}(v) = D^k(v)$, for all vertices v not adjacent to b.

5. If $k < n$ return to step 3, else go to step 6.

6. When $k = n$, we are done considering all the vertices. We are back to the original cut, since we have swapped all the vertices, i.e. roles of S and T are now reversed. Thus we have $\sum_{i=1}^{n} g(i) = 0$.

7. Examine the cumulative function $G(k) = \sum_{i=1}^{k} g(i),\ k = 1, \ldots, n$, and see where it is a minimum. If $G(k) \geq 0$, $k = 1, \ldots, n$ we stop. Else choose k such that $G(k)$ is most negative (break ties arbitrarily), and perform the k first interchanges. This will give a cut, which is better than what we started with.

8. We can repeat this procedure, until it runs out of steam. In our implementation we repeat the procedure 5 times in all.

The advantage of the Kernighan-Lin scheme is that by allowing a few positive $g(i)$'s in the beginning, we can get a $G(i)$ which is fairly negative, and we may able to escape from a local minimum if any.

We are now ready to sketch a preliminary SDP cutting plane approach for the maxcut problem. We will highlight some of the advantages of this approach over the LP cutting plane approach discussed in the previous section. We will also mention some of the potential difficulties in this approach along the way.

An SDP cutting plane approach for maxcut

1. **Initialize** Our initial SDP relaxation is the Goemans and Williamson SDP relaxation for maxcut (4.14).

2. **Solve the current SDP relaxation**, using a primal dual interior point method. This gives an upper bound on the optimal value of the maxcut problem.

3. **Separation :** Check all violated odd cycle inequalities. Bucket sort the resulting violated inequalities, and add a subset of constraints to the relaxation.

4. **Primal heuristic :** Use the Goemans-Williamson randomized rounding procedure to find a good incidence cut vector. This is a lower bound on the optimal value.

5. **Check for termination :** If the difference between the upper bound and the value of the best cut is less than an integer, STOP with optimality. If no violated cycle inequalities are found, and the difference is large, use branch and bound to complete the solution.

6. **Loop :** return to step 2

Assuming that we do not resort to branch and bound, at the termination of the cutting plane scheme we are solving the following problem (4.19)

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \text{diag}(X) & = & e, \\ & \sum_{ij \in C \setminus F} X_{ij} - \sum_{ij \in F} X_{ij} & \leq & |C| - 2, \\ & \text{C a cycle}, \ F \subseteq C, & & |F| \text{ odd}, \\ & X & \succeq & 0 \end{array} \tag{4.19}$$

Advantages of an SDP over an LP cutting plane approach

1. As we mentioned earlier the success of a cutting plane approach hinges on the choice of an initial relaxation of the combinatorial optimization problem. The Goemans and Williamson SDP relaxation (4.14) provides a provably good initial relaxation for the maxcut problem, with its use in generating a 0.878 approximation algorithm for the maxcut problem. This is in sharp contrast to the LP approach, where we can only guarantee a $\frac{1}{2}$ approximation algorithm for the initial LP relaxation (4.6). If we add in all the triangle inequalities, this approximation ratio improves to $\frac{2}{3}$.

2. The SDP relaxation (4.19) is tighter than an LP approach of optimizing over the odd cycle polytope. This is because the feasible region of (4.19) is contained within the odd cycle polytope, since the additional $-1 \leq X_{ij} \leq 1$

inequalities, that together with the odd cycle inequalities (4.17) describe the odd cycle polytope, are implied by $\mathrm{diag}(X) = e$, and $X \succeq 0$.

3. The Goemans and Williamson rounding procedure (step 3 in the approach enunciated above) provides a way to generate incidence vectors, which provide good lower bounds. We can improve these vectors by using the rounding procedure in conjunction with the Kernighan-Lin heuristic.

Unfortunately there is no free lunch, and the SDP approach presents its own share of difficulties. The important ones are mentioned below.

Potential difficulties in an SDP cutting plane scheme

1. The main tools for solving SDP's are interior point methods. Interior point methods have several advantages, chief among these their ability to solve a SDP to any degree of precision in a polynomial number of arithmetic operations. However they are still fairly limited in the size of problems they can handle. Problems with more than 3000 constraints are considered quite hard. Besides, interior point methods do not exploit any structure in the problem.

2. Secondly, and more importantly there is no convincing warm start strategy that allows us to re-optimize a slightly perturbed version of the original problem, after the addition of cutting planes, quickly. This is in contrast with an LP approach, where we could use the dual simplex method for re-optimization. Unfortunately there is no natural analog of the simplex method for the SDP, since the problem is no longer combinatorial, due the smoothness of the SDP feasible region. This can be partially remedied by Pataki's [101] simplex like method for the SDP.

With regard to point 1, recently Helmberg [46] has incorporated the spectral bundle approach (which as we discussed in the previous chapters is designed to solve large scale SDP's quickly) in a cutting plane approach to solving the maxcut problem.

As regards point 2, we discuss a number of warm start strategies for the maxcut problem in section 4.4.2.

4.4.1 Branch and bound in the SDP context

We provide a short writeup on branch and bound within the SDP context in this section. This will be useful when we discuss the incorporation of branch and bound in our cutting plane approach in section (4.6.4). Some excellent references for branch and bound within the SDP context of the maxcut problem are Helmberg and Rendl [53], and Mitchell [81].

We want to branch based on the values of $X_{ij} = (v_i^T v_j)$. Typically this is the most fractional variable, i.e. the X_{ij} closest to zero. The branching scheme is based on whether vertices i and j should be on the same side of the cut or on opposite sides. With this branching rule X_{ki} and X_{kj} are also then constrained to be either the same or different, $\forall k = \{1, \ldots, n\} \backslash \{i, j\}$. This means that the problem can be replaced by an equivalent semidefinite program of dimension one less. Without loss of generality let us assume that we are branching on whether vertices $n-1$ and n are on the same or opposite sides. Let we write the Laplacian matrix L in (4.14) as

$$L = \begin{bmatrix} \bar{L} & p_1 & p_2 \\ p_1^T & \alpha & \beta \\ p_2^T & \beta & \gamma \end{bmatrix}$$

Here $\bar{L} \in \mathcal{S}^{n-2}$, $p_1, p_2 \in \mathcal{R}^{n-2}$ and α, β, and $\gamma \in \mathcal{R}$. The SDP relaxation that corresponds to putting both $n-1$ and n on the same side is (4.20).

$$\begin{array}{lll} \max & \frac{1}{4} \begin{bmatrix} \bar{L} & p_1 + p_2 \\ p_1^T + p_2^T & \alpha + 2\beta + \gamma \end{bmatrix} \bullet X & \\ \text{s.t.} & \operatorname{diag}(X) = e & \\ \text{s.t.} & X \succeq 0 & \end{array} \tag{4.20}$$

with dual

$$\begin{array}{lrcl} \min & e^T y & & \\ \text{s.t.} & S & = & \mathrm{Diag}(y) - \frac{1}{4}\begin{bmatrix} \bar{L} & p_1 + p_2 \\ p_1^T + p_2^T & \alpha + 2\beta + \gamma \end{bmatrix} \\ & S & \succeq & 0 \end{array} \tag{4.21}$$

Note that $X, S \in \mathcal{S}^{n-1}$, and $y \in \mathcal{R}^{n-1}$, i.e. not only do we have a semidefinite program of dimension one less, but the number of constraints in (4.20) has dropped by one as well. This is because performing the same transformation (as the Laplacian) on the nth coefficient matrix $e_n e_n^T$ leaves it as $e_{n-1}e_{n-1}^T$, which is in fact the $n-1$th coefficient matrix.

On the other hand, putting $n-1$ and n on opposite sides, we get a similar SDP relaxation, with the Laplacian matrix now being $\frac{1}{4}\begin{bmatrix} \bar{L} & p_1 - p_2 \\ p_1^T - p_2^T & \alpha - 2\beta + \gamma \end{bmatrix}$

It is desirable that we use the solution to the parent node, in this case (4.14) to speed up the solution of the child (4.20). As we mentioned previously this is a major issue in the SDP, since there is no analogue to the dual simplex method, unlike the LP case for re-optimization. More details on this can be found in Mitchell [81].

Another important issue is determining good bounds for each of the subproblems, so that some of these subproblems in the branch and bound tree could be *fathomed*, i.e. not explicitly solved. In the LP approach, we can use reduced costs to estimate these bounds, and hence fix some of the variables without having to solve both subproblems. In the SDP case things are not so easy, since the constraints $-1 \leq X_{ij} \leq 1$ are not explicitly present in the SDP relaxation (they are implied instead through the $\mathrm{diag}(X) = e$ and $X \succeq 0$ constraints). Thus the dual variables corresponding to these constraints are not directly available. Helmberg [47] describes a number of approaches to fix variables in semidefinite relaxations.

4.4.2 Warm start strategies for the maxcut problem

In cutting plane algorithms it is of fundamental importance that re-optimization be carried out in reasonable time, after the addition of cutting planes. Since the oracle returned *deep* cutting planes, the current iterate $X^{current}$ is infeasible after the addition of cutting planes. Therefore we have to construct a new feasible point X^{start} for restarting the method.

In the following we assume that we add m odd cycle inequalities (4.17) to the initial relaxation (4.14). We express these linear constraints as $\mathcal{A}X + s = b$, where $b \in \mathcal{R}^m$, and $s \in {\mathcal{R}_+}^m$ is a vector of slack variables. Then the new semidefinite relaxation reads as

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \operatorname{diag}(X) & = & e \\ & \mathcal{A}(X) + s & = & b \\ & X & \succeq & 0 \\ & s & \geq & 0 \end{array} \tag{4.22}$$

with dual

$$\begin{array}{lrcl} \min & \sum_{i=1}^{n} y_i + \sum_{i=1}^{m} b_i y_{n+i} & & \\ \text{s.t.} & S & = & \operatorname{Diag}(y(1:n)) + \mathcal{A}^{\mathcal{T}} y(n+1:n+m) - \frac{L}{4} \\ & S & \succeq & 0 \\ & y(n+1:n+m) & \geq & 0 \end{array} \tag{4.23}$$

We first discuss two strategies of restarting the primal (4.22) since this is the more difficult problem, since all the cuts are *deep*. They are

1. **Backtracking along iterates** :

   This idea is originally due to Mitchell and Borchers [80] for the LP. The idea is to store all the previous iterates on the central path, during the course of solving the original SDP relaxation (4.14), and restart from the last iterate that is strictly feasible with respect to the new inequalities. Also, this point is hopefully close to the new central path; the interior point algorithm will work better if this is the case.

2. **Backtracking towards the analytic center** :

   This idea was employed in Helmberg and Rendl [53]. It is easy to see that the identity matrix I is the analytic center of GW feasible region. The idea then is to backtrack towards I along the straight line between the last iterate X^{prev} and I. Thus we choose $X^{start} = (\lambda X^{prev} + (1-\lambda)I)$ for some $\lambda \in [0,1)$. The matrix I is within the maxcut polytope (it can be expressed as a convex combination of incidence vectors of cuts); also it is some sort of a center for this polytope. To see this consider Theorem 24; averaging over all such matrices gives the matrix I. So it is guaranteed that the procedure will terminate with a strictly feasible primal iterate.

Restarting the dual (4.23) is a relatively straightforward manner, since we can get into the dual SDP cone $S \succeq 0$, by assigning arbitrarily large values to the first n components of y. In our approach, we assign all the new components of y the value 1, i.e. $y(n+1:n+m) = e$ (note that the new components of y are required to be positive, whereas the first n components are unrestricted in sign). Let $y^{prev} \in \mathcal{R}^n$ be the previous dual solution. Compute $\bar{S} = \mathrm{Diag}(y^{prev}) + \mathcal{A}^T e - \frac{L}{4}$, and estimate the magnitude of its most negative eigenvalue λ. $S^{start} = \bar{S} + \alpha\lambda I$, where $\alpha > 1$ is positive definite, and

$$y^{start} = \begin{bmatrix} y^{prev} + \lambda e(1:n) \\ e(n+1:n+m) \end{bmatrix}$$

in the usual MATLAB notation. Before we conclude we wish to address the following points in a cutting plane approach :

1. Since the separation oracle generates more violated inequalities than we are willing to include in our SDP relaxations, it is important that we select the promising ones from the vast set of violated inequalities. So far we have only considered the *amount* of violation, i.e. we add the most violated inequalities from the subset. Helmberg [45] considers the following : For a violated inequality compute the intersection of the straight line between the current iterate X_c and I, the analytic center of the primal feasible set. Then consider inequalities with a small distance to I.

2. With regard to dropping constraints, we drop those constraints whose dual variables are small. Another criterion is to use the ratio of the dual variable to the slack in the constraint, and drop constraints for which the ratio is small.

4.5 Large scale cutting plane approaches for maxcut

In this section we describe two intriguing recent approaches for the maxcut problem. These approaches could be incorporated in a cutting plane approach to solve the problem to optimality. We choose to highlight these approaches not only due to our interests; these approaches could potentially pave the way for large scale approaches for the maxcut problem. As we mentioned previously the SDP is a great theoretical tool in developing approximation algorithms for combinatorial optimization problems. However it is still a computational burden to solve large scale SDP's, i.e. with a large number of constraints.

A few points are now in order :

1. The first approach is due to Burer et al [21]. It is based on a rank 2 formulation of the maxcut problem. We discuss this in section 4.5.1

2. The second intriguing approach due to Muramatsu and Suzuki [88] replaces the GW SDP formulation (4.14) of the maxcut problem with a more tractable second order cone programming (SOCP) formulation (4.27). This approach is discussed in section (4.5.2).

4.5.1 Rank two relaxation heuristics for the maxcut problem

This approach is due to Burer et al [21]. We present a short overview below. The essential idea is to consider the following problem (4.24).

$$\begin{array}{lrcl} \max & \frac{L}{4} \bullet X & & \\ \text{s.t.} & \operatorname{diag}(X) & = & e \\ & X & \succeq & 0 \\ & \operatorname{rank}(X) & \leq & 2 \end{array} \tag{4.24}$$

From (4.11) it is clear that (4.24) is indeed a relaxation of the maxcut problem. It is tighter than the GW SDP relaxation (4.14), unless (4.14) solves the maxcut problem itself. A different way to visualize (4.24) is that in the maxcut formulation (4.8) the scalars x_i, $i = 1, \ldots, n$ are replaced by vectors $v_i \in \mathcal{R}^2$. Also these vectors v_i are unit vectors, i.e. $||v_i|| = 1$ since we require $x_i \in \{-1, 1\}$. Using polar coordinates, we can represent these unit vectors v_i, $i = 1, \ldots, n$ as

$$v_i = \begin{pmatrix} \cos\theta_i \\ \sin\theta_i \end{pmatrix}, \quad i = 1, \ldots, n$$

Let $\theta = \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_n \end{pmatrix} \in \mathcal{R}^n$. Using the fact that $v_i^T v_j = \cos(\theta_i - \theta_j)$, and defining the skew symmetric matrix $T_{ij}(\theta) = (\theta_i - \theta_j)$, we can rewrite (4.24) as

$$\min_{\theta \in \mathcal{R}^n} \quad \tfrac{1}{4} L \bullet \cos(T(\theta)) \tag{4.25}$$

This is an unconstrained optimization problem with a non-convex objective function. In general, it has multiple local and non-global minima. The gradient and the Hessian of the objective function are given below

$$\begin{aligned} g(\theta) &= [\tfrac{L}{2} \circ \sin(T(\theta))]^T e \\ H(\theta) &= W \circ \cos(T(\theta)) - \mathrm{Diag}([W \circ \cos(T(\theta))]e) \end{aligned}$$

The main computational task in the evaluation of the gradient and the Hessian is in computing the quantities $W \circ \cos(T(\theta))$ and $W \circ \sin(T(\theta))$. Burer et al [21] show that a local minimum to $f(\theta) = \frac{1}{4} L \bullet \cos(T(\theta))$ corresponds to finding a maximum cut in the graph.

Since the maximum cut problem is NP-hard, it is likely there are a number of saddle points of $f(\theta)$, and this necessitates a good minimization algorithm to solving (4.25). The authors also propose a deterministic rather than randomized procedure to round θ, obtained by solving (4.25), to a cut. This is another advantage of the rank two relaxation over the SDP relaxation in that it is inexpensive to examine all

possible cuts generated, making it easy to find the best one.

The major advantage of the approach is that we have only $O(n)$ variables, as opposed to $O(n^2)$ in (4.14). Thus we can handle larger problems. However it is not clear how we could incorporate this approach in a cutting plane approach; the method only serves as a good heuristic for the maxcut problem.

4.5.2 Second Order Cone Programming formulations of the maxcut problem

Recently second order cone programming (SOCP) approaches have attracted a lot of attention : Kim and Kojima [64] show through computational experiments that the SOCP relaxation is a reasonable compromise between the effectiveness of the SDP relaxation and the low computational cost of the LP relaxation.

Using similar ideas Muramatsu and Suzuki [88] proposed an SOCP relaxation for the maxcut problem. We present a short overview of their approach below.

In (4.10) we formulated the maxcut problem as an $\{-1, 1\}$ integer programming problem. Since $x_i \in \{-1, 1\}$ implies that $x_i^2 = 1$, we can rewrite the maxcut problem as minimizing a linear function subject to a set of quadratic constraints as follows

$$\begin{array}{lrcl} \max & w & & \\ \text{subject to} & w & \leq & x^T \frac{L}{4} x \\ & x^T e_j e_j^T x & \leq & 1 \\ & x^T (-e_j e_j^T) x & \leq & -1 \end{array} \tag{4.26}$$

Here we exploit the fact that maximizing $x^T \frac{L}{4} x$ is equivalent to maximizing w, where $w \leq x^T \frac{L}{4} x$. Also $x_j^2 = 1$ is rewritten as $x_j^2 \leq 1$ and $x_j^2 \geq 1$, which in turn can be rewritten as $x^T e_j e_j^T x \leq 1$ and $x^T (-e_j e_j^T) x \leq -1$ respectively.

If all the constraints in (4.26) were convex, we could formulate (4.26) as an SOCP (see Lobo et al [77] and Alizadeh and Goldfarb [2] for such a formulation).

Unfortunately the constraint $x^T (-e_j e_j^T) x \leq -1$ is non-convex (it is actually concave). We would need some ways to convexify this constraint. Muramatsu and

Suzuki [88] do this in the following manner. They consider vectors $u_{ij} = (e_i + e_j)$ and $v_{ij} = (e_i - e_j)$, $\forall\{i,j\} \in E$, and replace the non-convex quadratic constraint $x^T(-e_j e_j^T)x \leq 1$ with

$$\begin{aligned} x^T u_{ij} u_{ij}^T x - u_{ij} u_{ij}^T \bullet X \leq 0 \quad \forall\{i,j\} \in E \\ x^T v_{ij} v_{ij}^T x - v_{ij} v_{ij}^T \bullet X \leq 0 \quad \forall\{i,j\} \in E \end{aligned}$$

A quick note on why these constraints are valid : In arriving at a tractable relaxation of the maxcut problem we replace the rank one restriction on X, i.e. $X = xx^T$ with the convex constraint $X \succeq xx^T$. Now

$$X \succeq xx^T \quad \Leftrightarrow \quad C \bullet (X - xx^T) \geq 0, \forall C \succeq 0$$

Since the rank one matrices $u_{ij}u_{ij}^T$ and $v_{ij}v_{ij}^T$ are trivially positive semidefinite, they give rise to valid inequalities for the maxcut problem.

Since the constraints are now convex, we can rewrite the new problem as an SOCP. Muramatsu and Suzuki [88] exploit the structure of the maxcut problem to rewrite this SOCP in an efficient manner. We omit these details and proceed directly to their final relaxation

$$\begin{array}{lrcl} \max & -\sum_{\{i,j\}\in E} L_{ij} z_{ij} & & \\ \text{s.t.} & x_i^2 & \leq & 1, \quad i = 1, \ldots, n \\ & (x_i + x_j)^2 & \leq & s_{ij}, \quad \{i,j\} \in E \\ & (x_i - x_j)^2 & \leq & z_{ij}, \quad \{i,j\} \in E \\ & s_{ij} + z_{ij} & = & 4, \quad \{i,j\} \in E \end{array} \tag{4.27}$$

where $s_{ij} = u_{ij}^T X u_{ij}$ and $z_{ij} = v_{ij}^T X v_{ij}$ respectively. The number of variables in (4.27) is $O(m)$, where m is the number of edges in the graph. This is particularly useful if the graph is sparse.

Muramatsu and Suzuki [88] are able to include the triangle inequalities in their SOCP relaxation. It would be interesting to incorporate this SOCP relaxation in a cutting plane approach in the future.

4.6 An SDP method for the maxcut problem

This section deals with the major contribution of this chapter, which is a cutting plane SDP method for the maxcut problem.

The method is based on the semi-infinite formulation of the dual SDP relaxation (4.15). We described a cutting plane approach to solving this relaxation in the previous chapter. This cutting plane approach will be used as a subroutine to solve the SDP relaxations which arise while solving the maxcut problem. We describe our approach in section 4.6.2.

We must mention that Gruber and Rendl [43] have a similar approach, where they solve a semi-infinite formulation of the primal SDP relaxation (4.14). We must emphasize that they do not motivate it in this way; their aim is to strengthen the LP cutting plane approach by adding cutting planes valid for the SDP approach. An advantage of this approach is that we deal exclusively with the primal problem, and not oscillate between the primal and dual as in our approach. We present a short overview of their approach in section 4.6.1.

4.6.1 The Gruber Rendl approach : Work with the primal SDP

In this section we present a strengthened LP relaxation of the maxcut problem due to Gruber and Rendl [43]. This has a number of connections to our own LP approach discussed in section (4.6.2). We present a short overview of the scheme below.

Since the feasible region of (4.19) is contained within the odd cycle polytope, we can improve the relaxations in an LP cutting plane approach, by adding cutting planes $dd^T \bullet X \geq 0$, which force a matrix X to be positive semidefinite. We will relate this matrix X to the vector $x \in \mathcal{R}^m$ arising in the LP relaxations shortly.

For instance when we run out of odd cycle inequalities, we can strengthen our LP relaxations as follows : We solve the current LP relaxation to obtain $x \in \mathcal{R}^m$. Note that we are still dealing with the $\{0,1\}$ formulation of maxcut, so all the components of x are between 0 and 1. We then lift this vector x to the space $\mathcal{S}^n$ consisting of symmetric matrices X of size n using the affine transformation $X_{ij} = (1 - 2x_{ij}),\ \forall \{i,j\} \in E$. The diagonal entries of X are all set equal to

one. The dilemma is assigning values to the off diagonal entries of X that do not correspond to edges in G, so as to keep the overall matrix X psd.

This brings us to the *matrix completion problem*, a good account of which appears in Laurent [73]. We omit the essential details, but the idea is to examine every sub-matrix X_K of X, where $K \subseteq V$ is any clique in the graph, for positive semidefiniteness. In fact if the graph is chordal, then it is shown in Grone et al [41], that $X_K \succeq 0$, for each clique $K \subseteq V$ is both a necessary and sufficient condition for $X \succeq 0$.

Thus we examine X_K for any clique $K \subset V$ for positive semidefiniteness. If X_K is indefinite, then this shows that X is not psd either. The eigenvector v_K corresponding to the most negative eigenvalue of X_k gives rise to a cutting plane $v_K v_K^T \bullet X_k \geq 0$. This cutting plane can be lifted to the space of $\mathcal{S}^n$ by assigning zeros to those components of v_K that correspond to vertices not in the clique K. This gives rise to a cutting plane $v_K v_K^T \bullet X \geq 0$. We then project this inequality in $\mathcal{S}^n$ into the space $\mathcal{R}^m$ to give a valid cutting plane for the maxcut problem. Gruber and Rendl [43] also consider hypermetric inequalities in their LP approach; more details can be found in Gruber's thesis [43].

4.6.2 Our SDP approach : Work with the dual SDP

We first describe the entire algorithm in a nutshell. Each of these steps will be elaborated in more detail later.

1. **Initialize**

2. **Solve the dual maxcut SDP as an LP** using the interior point cutting plane scheme. Construct the primal matrix X.

3. **Generate good integer cut vectors** by running the Goemans-Williamson and Kernighan-Lin heuristics on the primal matrix X. Add these integer vectors into the LP relaxation, and resolve to find the new X.

4. **Check for termination :** If the difference between the upper bound, and the value of the best cut is small, STOP with optimality.

5. **Separation oracle :** Find violated odd cycle inequalities. Bucket sort the resulting violated inequalities, and add a subset of constraints to the relaxation. This changes the dual slack matrix S.

6. **Update the LP relaxation** by choosing the most important constraints in the LP relaxation, constraints based on the spectral factorization of X, the initial box constraints, and the best integer cut vector. Solve the LP relaxation.

7. **Loop :** return to step 2

We would like to emphasize a few points, before describing an iteration of the algorithm in detail.

1. We do not mention how accurately we solve the dual maxcut SDP as an LP in step 2. If we do not solve the SDP accurately enough, then the objective value of the LP relaxation is not guaranteed to be an upper bound on the maxcut value. In fact we typically have the following inequalities on the objective values of the various relaxations. The SDP here is (4.15), the dual relaxation for the maxcut problem.

$$\begin{array}{ccc} \text{Cheap LP relaxation to SDP} & \leq & \text{Optimal Maxcut value} \quad \leq \\ \text{Good LP relaxation to SDP} & \leq & \text{SDP objective value} \end{array}$$

2. The accuracy to which to which the SDP is solved as an LP affects the performance of the GW randomized rounding procedure. Using the same analysis as Goemans and Williamson [36], we can show that the value of the cut generated by the procedure is at least 0.878 times the value of the current LP relaxation, if all the edge weights are nonnegative, and $\frac{2}{\pi}$ when negative edge weights are allowed (such as those arising in Ising spin glass problems in statistical physics). However, as mentioned in point (1) we cannot always guarantee that the value of our LP relaxation is an upper bound on the maxcut value. Thus we do not have a performance guarantee out here. Nevertheless, we can say in a weak sense that the more accurately we solve the SDP as an LP in the cutting plane scheme, the more likely that a better cut is produced.

3. We add cutting planes in both the primal as well as the dual. We add cutting planes (4.17) based on the polyhedral structure of the maxcut polytope in the primal, while the cutting planes added in the dual are sometimes low dimensional faces of the dual SDP cone (in y space), and not always facets. (also refer to an earlier discussion in section 3.5.4).

4. As regards the primal, we cannot guarantee that our primal LP feasible region always contains the entire maxcut polytope; in most cases it does not. The primal LP feasible region increases, when we solve the dual SDP as an LP, (i.e. add cutting planes in the dual), and decreases, when we add cutting planes in the primal.

5. When we are adding cutting planes in the dual, we update the primal matrix $X = \sum_{i=1}^{m} x_i d_i d_i^T$. On the other hand, when we add cutting planes in the dual we are updating the dual slack matrix $S = \mathrm{Diag}(y) + \sum_{i=1}^{k} y_{n+i} A_i - \frac{L}{4}$.

We are now ready to describe an iteration of the algorithm in detail :

1. **Initialization** The initial dual LP relaxation is

$$\begin{array}{lllll} \min & \sum_{i=1}^{n} y_i & & (LDR_0) & \\ \text{s.t.} & y_i & \geq & \frac{L_{ii}}{4}, & i = 1, \ldots, n \end{array}$$

An optimal solution is $(x^0, y^0, z^0) = (e, \mathrm{diag}(\frac{L}{4}), 0) \in \mathcal{R}^n \times \mathcal{R}^n \times \mathcal{R}^n$. This corresponds to initial matrices $X^0 = I$, and $S^0 = \frac{L}{4} - \mathrm{Diag}(\frac{L}{4})$. Also $m^0 = n^0 = n$, and $p^0 = 0$ (these terms are defined below).

2. **The $(k+1)$th iteration of the algorithm**

   - In the succeeding discussion we refer to the LP relaxation of the dual SDP as the dual, and its dual as the primal. We fix some notation here : $m^k(n^k)$ are the number of dual (primal) constraints in the kth iteration. Let $\bar{d}^k(\bar{p}^k)$ be the number of dual (primal) constraints added in iteration k. Also, $\bar{d}^k = \sum_{j=1}^{e} \bar{d}^{kj}$, and $\bar{p}^k = \sum_{j=a}^{e} \bar{p}^{kj}$ (here $a, \ldots, e$ refer to the each of

the five stages in each iteration). We then use $d^k = \sum_{i=1}^{k} \bar{d}^i (p^k = \sum_{i=1}^{k} \bar{p}^i)$ to denote the number of dual (primal) constraints added from iterations 1 through k. Finally, $m^k = n + d^k$, and $n^k = n + p^k$ respectively.

- The dual LP relaxation from the kth iteration is (LDR_k)

$$
\begin{aligned}
\min \quad & \textstyle\sum_{i=1}^{n} y_i + \sum_{i=1}^{p^k} (|C_i| - 2) y_{n+i} \\
\text{s.t.} \quad & \begin{bmatrix} I & 0 & \dots & 0 \\ d_{n+1}.^2 & d_{n+1}^T A_1 d_{n+1} & \dots & d_{n+1}^T A_{p^k} d_{n+1} \\ \ddots & \vdots & \ddots & \vdots \\ d_{m^k}.^2 & d_{m^k} A_1 d_{m^k} & \dots & d_{m^k} A_{p^k} d_{m^k} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ y_{n+1} \\ \vdots \\ y_{n^k} \end{bmatrix} \\
& \geq \begin{bmatrix} \text{diag}(\frac{L}{4}) \\ d_{n+1}^T \frac{L}{4} d_{n+1} \\ \vdots \\ d_{m^k}^T \frac{L}{4} d_{m^k} \end{bmatrix} \qquad (LDR_k) \\
& y_{n+i} \ \geq \ 0, \quad i = 1, \dots, p^k
\end{aligned}
$$

and the primal relaxation (LPR_k) is

$$
\begin{aligned}
\max \quad & \tfrac{L}{4} \bullet (\textstyle\sum_{i=1}^{m^k} x_i d_i d_i^T) & & & \\
\text{s.t.} \quad & \text{diag}(\textstyle\sum_{i=1}^{m^k} x_i d_i d_i^T) & = & e & \qquad (LPR_k) \\
& A_i \bullet (\textstyle\sum_{i=1}^{m^k} x_i d_i d_i^T) & \leq & (|C_i| - 2), \quad i = 1, \dots, p^k & \\
& x_j & \geq & 0, \qquad j = 1, \dots, m^k &
\end{aligned}
$$

An odd cycle inequality (4.17) is written as $A \bullet X \leq (|C| - 2)$, where A is a symmetric matrix, and $|C|$ is the cardinality of the cycle. Also x_i is the primal variable corresponding to the ith dual constraint.

- (x^k, y^k, z^k) is the current solution. Here $x^k \in \mathcal{R}^{m^k}$, $y^k \in \mathcal{R}^{n^k}$, and $z^k \in \mathcal{R}^{m^k}$.

There are five ways in which the LP formulation is modified in every iteration :-

(a) **Improve the LP relaxation of the SDP using a cutting plane LP approach** Currently (LDR_k) is a relaxation of the dual SDP (SDD_k).

$$\begin{array}{lrcl} \min & \sum_{i=1}^{n} y_i + \sum_{i=1}^{p^k}(|C_i|-2)y_{n+i} & (SDD_k) & \\ \text{s.t.} & \text{Diag}(y(1:n)) + \sum_{i=1}^{p^k} A_i y_{n+i} - S & = & \frac{L}{4} \\ & S & \succeq & 0 \end{array}$$

We tighten the relaxation using an interior LP approach for the SDP as discussed in the previous chapter. To do this we add $\bar{d}^{k+1_a}$ cutting planes in the dual (LDR_k), which try and force $S^k = \text{Diag}(y(1:n)) + \sum_{i=1}^{p^k} A_i y_{n+i} - \frac{L}{4}$ to be psd. Carry out the updates (in that order) :

$$\begin{aligned} d^{k+1_a} &= d^k + \bar{d}^{k+1_a} \\ m^{k+1_a} &= n + d^{k+1_a} \end{aligned}$$

Also, no constraints are added in the primal, i.e. $\bar{p}^{k+1_a} = 0$. Thus $n^{k+1_a} = n^k$. The new LP relaxations are :

$$
\begin{array}{ll}
\min & \sum_{i=1}^{n} y_i + \sum_{i=1}^{p^k} (|C_i| - 2) y_{n+i} \\
\text{s.t.} & \begin{bmatrix}
I & 0 & \dots & 0 \\
d_{n+1}.^2 & d_{n+1}^T A_1 d_{n+1} & \dots & d_{n+1}^T A_{p^k} d_{n+1} \\
\ddots & \vdots & \ddots & \vdots \\
d_{m^k}.^2 & d_{m^k} A_1 d_{m^k} & \dots & d_{m^k} A_{p^k} d_{m^k} \\
d_{m^k+1}.^2 & d_{m^k+1} A_1 d_{m^k+1} & \dots & d_{m^k+1} A_{p^k} d_{m^k+1} \\
\ddots & \vdots & \ddots & \vdots \\
d_{m^{k+1_a}}.^2 & d_{m_{k+1_a}} A_1 d_{m^{k+1_a}} & \dots & d_{m^{k+1_a}} A_{p^k} d_{m^{k+1_a}}
\end{bmatrix}
\begin{bmatrix} y_1 \\ \vdots \\ y_n \\ y_{n+1} \\ \vdots \\ y_{n^k} \end{bmatrix} \\
& \geq \begin{bmatrix}
\operatorname{diag}(\frac{L}{4}) \\
d_{n+1}^T \frac{L}{4} d_{n+1} \\
\vdots \\
d_{m^k}^T \frac{L}{4} d_{m^k} \\
d_{m^k+1}^T \frac{L}{4} d_{m^k+1} \\
\vdots \\
d_{m^{k+1_a}} \frac{L}{4} d_{m^{k+1_a}}
\end{bmatrix} \qquad (LDR_{k+1_a}) \\
& y_{n+i} \geq 0, \quad i = 1, \dots, p^k
\end{array}
$$

with primal

$$
\begin{array}{lrcll}
\max & \frac{L}{4} \bullet (\sum_{i=1}^{m_a^{k+1}} x_i d_i d_i^T) & & & \\
\text{s.t.} & \operatorname{diag}(\sum_{i=1}^{m_a^{k+1}} x_i d_i d_i^T) & = & e & (LPR_{k+1_a}) \\
& A_i \bullet (\sum_{i=1}^{m_a^{k+1}} x_i d_i d_i^T) & \leq & (|C_i| - 2), \quad i = 1, \dots, p^k & \\
& x_j & \geq & 0, \quad j = 1, \dots, m_a^{k+1} &
\end{array}
\tag{4.28}
$$

Let $(x^{k+1_a}, y^{k+1_a}, z^{k+1_a}) \in \mathcal{R}^{m^{k+1_a}} \times \mathcal{R}^{n^k} \times \mathcal{R}^{m^{k+1_a}}$ be a solution to (LDR_{k+1_a}) and (LPR_{k+1_a}) respectively.

(b) **Improve the LP relaxations by adding good cut vectors if necessary** Compute $X^{k+1_a} = \sum_{i=1}^{m_a^{k+1}} {x_i}^{k+1_a} d_i d_i^T$. From Theorem 9 we know that X^{k+1_a} is feasible in the primal SDP. In other words X^{k+1_a} is psd with

diagonal entries one, and strictly satisfies the current odd cycle inequalities. Run the Goemans and Williamson randomized rounding procedure on X^{k+1_a}, and generate a number of good cut vectors. As we remarked earlier, the strength of these cuts depends on how accurately we solve (SDD_k) as an LP in the cutting plane scheme. We then improve the cuts using the Kernighan and Lin (KL) heuristic. If the objective value of these cuts is better than the current LP relaxation, then add the 5 best cuts as cutting planes in the dual (LDR_{k+1_a}) Carry out the updates (in that order)

$$\begin{aligned} \bar{d}^{k+1_b} &= 5 \\ d^{k+1_b} &= d^{k+1_a} + \bar{d}^{k+1_b} \\ m^{k+1_b} &= n + d^{k+1_b} \end{aligned}$$

Also, since no constraints are added in the primal $n^{k+1_b} = n^k$. The new LP relaxations are

$$\begin{aligned} \min \quad & \textstyle\sum_{i=1}^{n} y_i + \sum_{i=1}^{p^k} (|C_i| - 2) y_{n+i} \\ \text{s.t.} \quad & \begin{bmatrix} I & 0 & \dots & 0 \\ d_{n+1}.^2 & d_{n+1}^T A_1 d_{n+1} & \dots & d_{n+1}^T A_{p^k} d_{n+1} \\ \ddots & \vdots & \ddots & \vdots \\ d_{m^{k+1_b}}.^2 & d_{m^{k+1_b}} A_1 d_{m^{k+1_b}} & \dots & d_{m^{k+1_b}} A_{p^k} d_{m^{k+1_b}} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_{n^k} \end{bmatrix} \\ & \geq \begin{bmatrix} \operatorname{diag}(\frac{L}{4}) \\ d_{n+1}^T \frac{L}{4} d_{n+1} \\ \vdots \\ d_{m^{k+1_b}}^T \frac{L}{4} d_{m^{k+1_b}} \end{bmatrix} \qquad (LDR_{k+1_b}) \\ & y_{n+i} \geq 0, \quad i = 1, \dots, p^k \end{aligned}$$

and

$$\begin{aligned} \max \quad & \tfrac{L}{4} \bullet (\textstyle\sum_{i=1}^{m^{k+1_b}} x_i d_i d_i^T) \\ \text{s.t.} \quad & \operatorname{diag}(\textstyle\sum_{i=1}^{m^{k+1_b}} x_i d_i d_i^T) = e \qquad (LPR_{k+1_b}) \\ & A_i \bullet (\textstyle\sum_{i=1}^{m^{k+1_b}} x_i d_i d_i^T) \leq (|C_i| - 2), \quad i = 1, \dots, p^k \\ & x_j \geq 0, \quad j = 1, \dots, m^{k+1_b} \end{aligned}$$

Let $(x^{k+1_b}, y^{k+1_b}, z^{k+1_b}) \in \mathcal{R}^{m^{k+1_b}} \times \mathcal{R}^{n^k} \times \mathcal{R}^{m^{k+1_b}}$ be a solution. Recompute the primal matrix, $X^{k+1_b} = \sum_{i=1}^{m^{k+1_b}} {x_i}^{k+1_b} d_i d_i^T$.

(c) **Find odd cycle inequalities violated by the current solution** Feed X^{k+1_b} to the Barahona-Mahjoub separation oracle which returns violated odd cycle inequalities (4.17), which are facets of the maxcut polytope. Bucket sort these odd cycle inequalities by violation, and add the $\bar{p}^{k+1_c}$ most violated ones to (LPR_{k+1_b}). Carry out the updates

$$\begin{aligned} p^{k+1_c} &= p^k + \bar{p}^{k+1_c} \\ n^{k+1_c} &= n + p^{k+1_c} \\ S^{k+1_c} &= \mathrm{Diag}(y(1:n)) + \sum_{i=1}^{p^{k+1_c}} {y_{n+i}}^{k+1_c} A_i - \tfrac{L}{4} \end{aligned}$$

Since no constraints are added in the dual in this stage, we have $m^{k+1_c} = m^{k+1_b}$. The new LP relaxations are

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} y_i + \sum_{i=1}^{p^{k+1_c}} (|C_i| - 2) y_{n+i} \\ \text{s.t.} \quad & \begin{bmatrix} I & 0 & \dots & 0 \\ {d_{n+1}.}^2 & d_{n+1}^T A_1 d_{n+1} & \dots & d_{n+1}^T A_{p^{k+1_c}} d_{n+1} \\ \ddots & \vdots & \ddots & \vdots \\ {d_{m_c^{k+1}}.}^2 & d_{m_c^{k+1}} A_1 d_{m_c^{k+1}} & \dots & d_{m_c^{k+1}} A_{p^{k+1_c}} d_{m_c^{k+1}} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_{n^k} \\ y_{n^k+1} \\ \vdots \\ y n^{k+1_c} \end{bmatrix} \\ & \geq \begin{bmatrix} \mathrm{diag}(\frac{L}{4}) \\ d_{n+1}^T \frac{L}{4} d_{n+1} \\ \vdots \\ d_{m^{k+1_c}}^T \frac{L}{4} d_{m^{k+1_c}} \end{bmatrix} \qquad (LDR_{k+1_c}) \\ & y_{n+i} \geq 0, \quad i = 1, \dots, p^{k+1_c} \end{aligned}$$

and

$$\begin{array}{lrcll} \max & \frac{L}{4} \bullet (\sum_{i=1}^{m_b^{k+1}} x_i d_i d_i^T) & & & \\ \text{s.t.} & \text{diag}(\sum_{i=1}^{m_b^{k+1}} x_i d_i d_i^T) & = & e & (LPR_{k+1_c}) \\ & A_i \bullet (\sum_{i=1}^{m_b^{k+1}} x_i d_i d_i^T) & \leq & (|C_i| - 2), & i = 1, \ldots, p^{k+1_c} \\ & x_j & \geq & 0, & j = 1, \ldots, m^{k+1_c} \end{array}$$

(d) **Drop constraints in the dual LP that no longer appear important** The rule is to drop those constraints, whose primal variables x_i are small. From Corollary 5 we have $\sum_{i=1}^{m^{k+1_b}} {x_i}^{k+1_b} = n$. (since trace$(X) = n$), where n is the number of nodes in the graph). We are implicitly assuming that all the d's are normalized, else we replace x_i with $x_i(d_i^T d_i)$. Constraints with a small value of ${x_i}^{k+1_b}$ in relation to n can be safely dropped without adversely affecting the LP relaxation. In most instances, the value of the LP relaxation is determined entirely by the value of the best cut. In this case, the LP is degenerate, and all the constraints with zero ${x_i}^{k+1_b}$ values can be dropped. We sort constraints $n+1, \ldots, n+d^{k+1_c}$ based on increasing values of their dual variables ${x_i}^{k+1_b}$, $i = n+1, \ldots, n+d^{k+1_c}$, and retain $q = O(\sqrt{n^{k+1_c}})$ constraints in all (see Theorem 16). We ensure that best integer vector is among these q constraints. The relaxation (LDR_{k+1_d}) has the following constraints :

- The box constraints, i.e. the first n constraints in (LDR_{k+1_c}).
- The q best constraints from the earlier relaxation (LDR_{k+1_c}) (see above).
- Compute a spectral factorization $X^{k+1_b} = P \Lambda P^T$, and add in the eigenvectors p_i, $i = 1, \ldots, r$ corresponding to nonzero eigenvalues of X^{k+1_b} (r here is the rank of X^{k+1_b}). In a sense, the eigenvectors corresponding to the nonzero eigenvalues *aggregate* the important information contained in the constraints added so far.

Let

$$X_{k+1_d} \quad = \quad (\sum_{i=1}^{n} x_i e_i e_i^T + \sum_{i=1}^{q} x_{n+i} d_{n+i} d_{n+i}^T + \sum_{i=1}^{r} x_{n+q+i} p_i p_i^T)$$

The final LP relaxations are

$$
\begin{array}{ll}
\min & \sum_{i=1}^{n} y_i + \sum_{i=1}^{p^{k+1_d}} (|C_i| - 2) y_{n+i} \\
\text{s.t.} & \begin{bmatrix}
I & 0 & \dots & 0 \\
d_{n+1}.^2 & d_{n+1}^T A_1 d_{n+1} & \dots & d_{n+1}^T A_{p^{k+1_d}} d_{n+1} \\
\ddots & \vdots & \ddots & \vdots \\
d_{n+q}.^2 & d_{n+q}^T A_1 d_{n+q} & \dots & d_{n+q}^T A_{p^{k+1_d}} d_{n+q} \\
p_1.^2 & p_1^T A_1 p_1 & \dots & p_1^T A_{p^{k+1_d}} p_1 \\
\ddots & \vdots & \ddots & \vdots \\
p_r.^2 & p_r^T A_1 p_r & \dots & p_r^T A_{p^{k+1_d}} p_r
\end{bmatrix}
\begin{bmatrix} y_1 \\ \vdots \\ y_{n^k} \\ y_{n^k+1} \\ \vdots \\ y n^{k+1_d} \end{bmatrix} \\
& \geq \begin{bmatrix}
\mathrm{diag}(\frac{L}{4}) \\
d_{n+1}^T \frac{L}{4} d_{n+1} \\
\vdots \\
d_{n+q}^T \frac{L}{4} d_{n+q} \\
p_1^T \frac{L}{4} p_1 \\
\vdots \\
p_r^T \frac{L}{4} p_r
\end{bmatrix} \qquad (LDR_{k+1_d}) \\
& y_{n+i} \geq 0, \quad i = 1, \dots, p^{k+1_d}
\end{array}
$$

and

$$
\begin{array}{lrcll}
\max & \frac{L}{4} \bullet X_{k+1_d} & & & \\
\text{s.t.} & \mathrm{diag}(X_{k+1_d}) & = & e & (LPR_{k+1_d}) \\
& A_i \bullet X_{k+1_d} & \leq & (|C_i| - 2), & i = 1, \dots, p^{k+1_d} \\
& x_j & \geq & 0, & j = 1, \dots, (n+q+r)
\end{array}
$$

We also have

$$
\begin{aligned}
d^{k+1_d} &= q + r \\
m^{k+1_d} &= n + d^{k+1_d}
\end{aligned}
$$

and $p^{k+1_d} = p^{k+1_c}$, and $n^{k+1_d} = n^{k+1_c}$.

(e) **Restart the LP relaxations after adding odd cycle inequalities** We need to exploit warm start information in solving (LDR_{k+1_d}) and (LPR_{k+1_d}) respectively. Since the odd cycle inequalities returned in

stage c, are designed to cut off the primal matrix X^{k+1_b}, the old solution $(x^{k+1_b}, y^{k+1_b}, z^{k+1_b})$ is no longer strictly feasible in (LDR_{k+1_d}) and (LPR_{k+1_d}). Consider the primal LP relaxation first.

Add positive linear combinations of $e_i e_i^T$, $i = 1, \ldots, n$ to

$$\bar{X}^{k+1_e} = \tfrac{1}{2}\{\textstyle\sum_{i=1}^{q} x_i{}^{k+1_d} d_{n+i} d_{n+i}^T + \sum_{i=1}^{r} \lambda_i p_i p_i^T\}$$

until $X^{k+1_e} = \sum_{i=1}^{n} \alpha_i e_i e_i^T + \bar{X}^{k+1_e}$ satisfies the equality constraints, i.e. $\mathrm{diag}(X^{k+1_e}) = e$. To strictly satisfy the odd cycle inequalities $A_i \bullet X \leq (|C_i| - 2)$, $i = 1, \ldots, p^{k+1_d}$, we backtrack towards I along the straight line between X^{k+1_e} and I (also see section 4.4.2). Since I is the analytic center of the feasible region $\{X : \mathrm{diag}(X) = e, X \succeq 0\}$, which contains the max cut polytope, a line search in this direction should enable us to strictly satisfy the violated odd cycle inequalities. The final primal slack matrix X is : $X^{k+1} = \sum_{i=1}^{m^{k+1}} \bar{x}_i^{k+1} d_i d_i^T$, with $\bar{x}^{k+1}$ strictly feasible in (LPR_{k+1_d}).

As regards the dual, perturb S^{k+1_b} until it is psd. Let λ be the magnitude of the most negative eigenvalue of S^{k+1_b}. Denote the constraint matrix and rhs in (LDR_{k+1_d}) as A^{k+1_d} and b^{k+1_d}. Compute

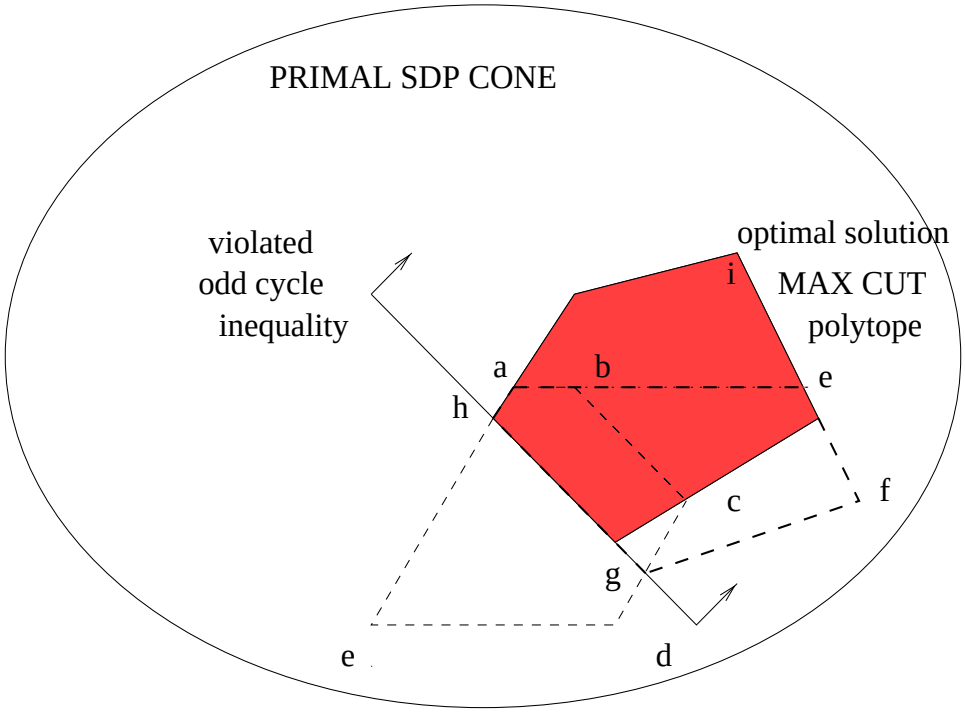
$$\begin{aligned} \bar{y}^{k+1} &= y^{k+1_b} + \lambda e \\ \bar{s}^{k+1} &= A^{k+1_d} \bar{y}^{k+1} - b^{k+1_d} \end{aligned}$$

$(\bar{x}^{k+1}, \bar{y}^{k+1}, \bar{s}^{k+1})$ is the desired strictly feasible starting point. Solve (LDR_{k+1_d}) and (LPR_{k+1_d}) with this starting point to obtain $(x^{k+1}, y^{k+1}, s^{k+1})$

3. Set $k = k + 1$ and return to step 2.

The overall SDP cutting plane method for maxcut is best summarized in Figure 4.3. The figure illustrates how the primal LP feasible region evolves during the course of the algorithm. Since we are solving a *constrained* version of the primal SDP relaxation, the LP feasible region is always within the corresponding primal SDP cone. The polytope shaded in red is the maxcut polytope. Suppose we are in

the kth iteration, and the LP feasible region is the polytope $abcdea$. Since we have solved the current dual SDP only approximately as an LP, the polytope $abcdea$ does not contain the entire maxcut polytope. The Barahona separation oracle returns an odd cycle inequality, that is violated by the current LP solution x^{k+1}. The new LP feasible region is now the polytope $abcgha$. We then try and grow this LP feasible region, by adding cuts in the dual to try and force the new dual slack matrix S^{k+1} to be positive semidefinite. At the end of the process, let $abefgh$ be the new LP feasible region. Although, $abefgha$ does not contain the maxcut polytope either, it is a better approximation than the earlier polytope $abcdea$. Proceeding in this manner, we eventually hit the optimal vertex i.



abcde : Initial LP relaxation
abcgh : LP region after adding violated odd cycles
abefgh : after the interior point sdp cutting plane scheme
Max Cut Polytope : Shaded in red
i : Optimal integer cut vector

Figure 4.3: The cutting plane SDP method for the maxcut problem

4.6.3 Convergence of the scheme

In this section we present a short note on the convergence of the scheme. A few points are already in order :

1. The interior point cutting plane approach for the SDP converges to the optimal SDP relaxation (as discussed in section 3.4 we can show that this occurs in a polynomial number of arithmetic operations, if we employ the 2 norm oracle, and central cuts).

2. The solution to the sequence of SDP relaxations need not converge to the optimal maxcut solution. When the approach terminates, we would have solved the SDP relaxation (4.19). The maxcut polytope may well be strictly within the feasible region of this relaxation (4.19), and in this case we can only guarantee an upper bound.

3. The overall solution to (4.19) can be carried out in polynomial time. This follows from item 1, and the fact that the Barahona-Mahjoub separation oracle can also be carried out in polynomial time.

4.6.4 Branch and bound in our approach

We investigate incorporating our approach in a branch and bound scheme in this section. For convenience, ease of exposition, and without loss of generality let us assume that we are solving the following problem (4.29), i.e. we assume that no odd cycle inequalities are present. We note that (4.29) is a semi-infinite formulation of the dual SDP relaxation of maxcut (4.15).

$$\begin{array}{lll} \min & e^T y & \\ \text{s.t.} & \sum_{i=1}^n d_{ji}^2 y_i & \geq \quad d_j^T \frac{L}{4} d_j, \quad j = 1, \ldots, m \end{array} \tag{4.29}$$

with dual

$$\begin{array}{lll} \max & \frac{L}{4} \bullet (\sum_{j=1}^m x_j d_j d_j^T) & \\ \text{s.t.} & \text{diag}(\sum_{j=1}^m x_j d_j d_j^T) \quad = \quad e & \\ & x_j \quad \geq \quad 0, \quad j = 1, \ldots, m & \end{array} \tag{4.30}$$

We resort to branch and bound. First we construct the primal matrix $X = \sum_{l=1}^{m} x_l d_l d_l^T$. We branch based on the values of $X_{ij} = \sum_{l=1}^{m} x_l d_{li} d_{lj}$. Since we have $-1 \leq X_{ij} \leq 1$, we branch on the X_{ij} closest to zero (the most fractional variable). We have two subproblems based on whether nodes i and j are on the same side, or opposite sides. Accordingly the variables X_{ik}, X_{jk} are constrained to be the same or different (opposite in sign). Thus we can reduce the dimension of the SDP by one. The Laplacian matrix $\frac{L}{4} \in \mathcal{S}^n$ changes to $\frac{\bar{L}}{4}$, as described in section (4.4.1). We wish to modify the vectors d_k, $k = 1, \ldots, m$, to exploit warm start information, as follows : Every d_k is now a vector $\bar{d_k}$ in $\mathcal{R}^{n-1}$ with $\bar{d_{kl}} = d_{kl}$, $l = \{1, \ldots, n\} \backslash \{i, j\}$, and $\bar{d_{ki}} = \frac{1}{2}(d_{ki} + d_{kj})$ if nodes i and j were on the same side previously, and $\bar{d_{ki}} = \frac{1}{2}(d_{ki} - d_{kj})$ otherwise. To see this, note that we wish to achieve $\bar{d}^T \bar{S} \bar{d} = d^T S d$, resulting in the above choice whenever $d_i = d_j$ or $d_i = -d_j$. We also add cutting planes which try and force $S = \mathrm{Diag}(y) - \frac{\bar{L}}{4} \in \mathcal{S}^{n-1}$ to be psd.

The branch and bound approach should also help in lowering the upper bounds in the cutting plane approach.

4.7 Computational results

In this section we report our preliminary computational experience. All results were obtained on a *Sun Ultra 5.6, 440 MHz* machine with *128 MB* of memory. We carried out our tests on a number of maxcut problems from *SDPLIB* [17], the *DIMACS Implementation Challenge* [102], and some randomly generated Ising spin glass problems, i.e. maxcut problems with ± 1 edge weights from Mitchell [82]. We compare our results with a traditional SDP cutting plane scheme that uses SDPT3 (Toh et al [114]) to solve the SDP relaxations.

We implemented our entire approach within the *MATLAB* environment. The LP solver was Zhang's *LIPSOL* [121]; we suitably modified the algorithm to restart the LP relaxations with a strictly feasible starting point. The SDP separation oracle discussed in the previous chapter employs the MATLAB Lanczos solver for the 2 norm, for the ∞ norm we use Vanderbei's LOQO [116]. The Barahona-Mahjoub separation oracle uses the *CPLEX* [25] network optimization solver to solve the shortest path problem as an LP.

The computational results are summarized in Table 4.1. The columns in the table represent

Name Problem name

n Number of nodes

m Number of edges

Maxcut Objective value of the best incidence vector

SDP1 Objective value over the elliptope (4.14)

SDP2 Objective value over the elliptope and odd cycles (4.19)

UB The upper bound returned by the cutting plane scheme

Name	n	m	Maxcut(Opt) [4]	SDP1 [5]	SDP2 [6]	UB [7]
gpp100 [1]	100	269	210	221.69	211.27	212.56
gpp1241 [1]	124	149	137	141.91	137	137.45
gpp1242 [1]	124	318	256	269.64	257.66	258.60
gpp1243 [1]	124	620	446	467.68	458.35	458.90
gpp1244 [1]	124	1271	834	864.26	848.92	
gpp2501 [1]	250	331	305	317.23	305	307.83
gpp2502 [1]	250	612	502	531.78	511.12	520.19
gpp5001 [1]	500	625	574	598.11	576.64 [8]	584.91
toruspm-8-50 [2]	512	1536	456	527.81	464.70 [8]	
torusg3-8 [2]	512	1536	412.75 (416.84)	457.36	417.69 [8]	428.18
ising10 [3]	100	200	70	79.22	70	70.70
ising20_1 [3]	400	800	282	314.25	282	288.04
ising20_2 [3]	400	800	268	305.63	268	271.82
ising30 [3]	900	1800	630 (632)	709.00	632	

Table 4.1: Test Results on Maxcut

[1]All results for 10 iterations of the SDP scheme
[1]SDPLIB
[2]DIMACS
[3]Random Ising Spin glass problems (Mitchell)
[4]Best integer solution from the SDP scheme (Optimal solution)
[5]Over the elliptope
[6]Over the elliptope and odd cycles
[7]Upper bounds from our scheme
[8]Best results obtained using an interior point approach

Each iteration of the cutting plane approach (steps 1 to 4 discussed in section 4.6.2) is faster than an interior point iteration, i.e. solve the SDP relaxations using a warm start procedure, and call the separation oracle to find violated odd cycle inequalities. This is especially true if the graph is sparse. We ran 10 cutting plane iterations in all. In our LP subroutine for the SDP, we add 5 cutting planes in each iteration, our starting tolerance is 1, and this is lowered by 0.9 in each iteration. Initially, we solve our SDP relaxations very cheaply, i.e. perform only 5 LP cutting plane iterations, and we increase this number if we are unable to improve the best incidence cut vector. Also, we add the 100 most violated odd cycle inequalities returned by the Barahona-Mahjoub approach in each iteration.

We must mention that our technique of estimating upper bounds performs badly. Therefore to obtain the upper bounds mentioned above, we solve the final primal SDP relaxation with the odd cycle inequalities to optimality using SDPT3 (Toh et al [114]). Interestingly, we were able to obtain the optimal integer solution in most cases (except problems torusg3-8 and ising30). The former problem is not easy, since the maxcut solution is not integer, whereas the latter problem is currently as big as we can possibly handle. For most of the problems we do not have a proof of optimality, since the SDP relaxation over the elliptope and odd cycles is not enough, and there is yet some gap involved.

To give an estimate of the times involved we performed the following experiment :

1. We ran our code against a standard SDP cutting plane scheme that employed $SDPT3$ [114] to solve the SDP relaxations. We chose $SDPT3$ over any other solver since it offered the flexibility for a warm start procedure.

2. We chose four problems from SDPLIB.

3. We ran 10 cutting plane iterations in all.

4. In our approach, we solve the SDP relaxations very cheaply (5 LP cutting plane iterations), and in every 5th iteration we solve this SDP relaxation more accurately (25 iterations). In $SDPT3$ we solve the SDP relaxations to a

moderate tolerance $1e-2$, and in very 5 the iteration we solve the SDP more accurately $(1e-6)$.

5. The other parameters for our LP cutting plane approach for the SDP are the same as mentioned above.

6. We add the $\frac{n}{4}$ most violated odd cycle ineqaulities returned by the Barahona-Mahjoub approach in each iteration.

The results can be found in table 4.2. A point to be noted here : We are using

Problem Name	SDPT3			Our approach		
	UB	LB	Time	UB	LB	Time
gpp100	212.26	210	321	216.86	210	485
gpp1241	137	137	197 [1]	138.33	137	488
gpp1242	258.4	256	423	263.73	256	620
gpp1243	458.86	446	409	461.92	446	832
gpp1244	851.01	834	468	857.35	834	902
gpp2501	305	305	2327	309.95	304	1073
gpp2502	512.06	502	1739	521.06	502	1687
gpp5001	-	-	-	583.65	574	2893

Table 4.2: Comparing two SDP cutting plane schemes for maxcut

an earlier version of $SDPT3$ namely $SDPT3-2.3$ to solve the SDP relaxations. This version does not handle linear blocks (the later version $SDPT3-3.1$ introduces this flexibility). Thus we increase the size of the semidefinite block by one, for every odd cycle inequality. The transformation is shown below.

$$A \bullet X \leq |C| - 2 \quad \Rightarrow \quad \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} X & 0 \\ 0 & x_s \end{bmatrix} = |C| - 2$$

This is clearly not very efficient procedure. However the main purpose of table (4.2) was show how our cutting plane approach scales with the problem size, and to compare it with a traditional SDP cutting plane approach. The interior point approach solved the problem gpp1241 to optimality in just 5 iterations. On the

[1] Converged in 5 iterations

other hand on problem gpp5001, the interior point method was unable to complete the stipulated 10 iterations.

Finally, we have not tested the branch and bound scheme as yet. More details can be found in the forthcoming paper Krishnan and Mitchell [71].

4.8 Conclusions

We have presented a cutting plane technique for approximating the maxcut problem. This linear approach allows one to potentially solve large scale problems. Here are some conclusions based on our preliminary computational results

1. It is worth reiterating that although our scheme behaves like an SDP approach for the maxcut problem, it is entirely a polyhedral LP approach. The SDP relaxations themselves are solved within a cutting plane scheme, so our scheme is really a cutting plane approach within another cutting plane approach.

2. The cutting plane approach is able to pick the optimal maxcut incidence vector quickly, but the certificate of optimality takes time.

3. Our technique for computing upper bounds performs poorly. We are currently working on a number of techniques to improve this upper bound based on using information from the eigenvector based on the most negative eigenvalue. We present some of this work in the chapter on forthcoming work. Nevertheless we believe that if we are able to handle this difficulty, then our approach can be effective on large scale maxcut problems.

4. The main computational task in our approach is the time taken by SDP separation oracle, described in the previous chapter. Another difficulty is that the oracle returns cutting planes that lead to dense LP relaxations. One way is to use the *matrix completion* idea as in Gruber and Rendl [43], and examine the positive semidefiniteness of S_K for any clique $K \subseteq V$ (here S_K is a sub-matrix of S corresponding to only those entries in the clique). This should speed up the separation oracle, which will also return sparser constraints, since any cutting plane for S_k can be lifted to $\mathcal{S}^n$ by assigning zero entries to the all components of the cutting plane not in K.

5. We are also trying to incorporate this approach in a branch and cut approach to solving the maxcut problem, i.e. we resort to branching when we are not able to improve our best cut values obtained so far, or our upper bounds. We discussed an approach in section (4.6.4). Some preliminary results can be obtained in the chapter for forthcoming work. Again, the major issue is how to use warm start information from the parent node, to re-optimize the child nodes quickly.

6. Helmberg [46] shows that better SDP relaxations can be obtained by enlarging the number of cycles in the graph, i.e. adding edges of weights 0 between non-adjacent nodes. This gives rise to a complete graph, and the only odd cycles now are triangle inequalities, which can be inspected by complete enumeration. Again, this in sharp contrast to the LP case, where adding these redundant edges does not improve the relaxation. We hope to utilize this feature in our cutting plane approach, in the near future.

7. We hope this extend this approach to other combinatorial optimization problems such as min bisection, k way equipartition, and max stable set problems based on a Lovasz theta SDP formulation.

CHAPTER 5
Concluding Remarks

"Do I contradict myself? Very well then I contradict myself."
— Walt Whitman (Song of Myself)

5.1 Summary of contributions

In this thesis, we have addressed several fundamental and algorithmic issues regarding linear programming (polyhedral) approaches to solving semidefinite programming problems. This gives us the flexibility to use the state of art linear solvers available. Also, we can incorporate these LP approaches to the SDP within a branch and cut framework to solving integer programming problems.

One of the various characterizations of the positive semidefiniteness constraint leads to a semi-infinite LP formulation for the SDP. We present a semi-infinite LP formulation for the dual SDP in Chapter 2, and discuss the issue of its discretization in detail. Using the notions of nondegeneracy and basic feasible solutions introduced in Anderson and Nash [6] for semi-infinite linear programs we are able to recover a theorem due to Pataki [99] on the rank of extreme matrices in SDP (this was first obtained in Alizadeh et al [4]). Moreover, this theorem implies that not more than $O(\sqrt{k})$ constraints are required in the LP relaxations. To generate these constraints we employ the spectral bundle method due to Helmberg and Rendl [52]. This scheme recasts any SDP with a bounded primal feasible set as an eigenvalue optimization problem, that can be tacked by bundle methods for nondifferentiable optimization. We present a short overview of their scheme, and then present the rationale for using the columns in the bundle P maintained by this approach, as our linear constraints. We demonstrate the effficiency of the approach on two combinatorial examples, namely the max cut and min bisection problems. Although the spectral bundle approach in itself solves the SDP, we wish to highlight two important features of our approach :

1. The LP approach provides an upper bound for the spectral bundle scheme

(when (SDP) is a minimization problem). Note that we are solving a relaxation of the dual; a maximization problem, whereas the spectral bundle always reports the objective value at a feasible y.

2. When the spectral bundle terminates, the primal problem is yet infeasible, since $\mathcal{A}(X) = b$ is attained only in the limit. Our scheme presents a way to generate a primal feasible matrix X. This is particularly important when the SDP is itself a relaxation of some underlying combinatorial optimization problem, and we want to solve this problem to optimality in a branch and cut SDP approach.

We then discuss a cutting plane LP approach for the SDP in Chapter 3. This scheme exploits the polynomial time separation oracle for the SDP. The worst case polynomial complexity of this approach is comparable to that of interior point methods for the SDP, and actually improves as the number of constraints k in the SDP go up, i.e. $k = O(n^2)$. To make the scheme competitive in practice with interior point methods for the SDP several refinements are necessary. In particular the cutting plane method requires solving the LP relaxations approximately using an interior point method, since this results in better cutting planes. We experiment with various separation oracles generating deep cuts, and techniques to restart the LP relaxations with strictly feasible starting points. We also relate these cutting planes to the geometry of the SDP cone, and finally test the approach on the maxcut SDP with encouraging results. We wish to emphasize the following points

1. One needs to fully exploit the tremendous flexibility available in these schemes.

2. The scheme is different from cutting plane LP approaches for integer programming problems, where facets of the underlying polytope are readily available. In the SDP we are at the mercy of an oracle that returns cutting planes, which are facets and sometimes lower dimensional faces of the SDP cone depending on the multiplicity of the smallest eigenvalue of the dual slack matrix S.

In chapter 4 we incorporate the cutting plane LP approach to the SDP in an SDP approach for the maxcut problem. In particular, we formulate the dual of the

well known SDP relaxation for maxcut as a semi-infinite LP, and solve it using an interior point cutting plane scheme. We then add cutting planes, which are facets of the maxcut polytope, to the primal problem in order to tighten the SDP relaxations. The approach resembles a SDP cutting plane scheme for the maxcut problem; in reality it uses an LP subroutine to solving the SDP within an overall LP cutting plane approach for integer programming. This overcomes another shortcoming of an SDP cutting plane scheme, where there is no convincing warm start strategy, that allows one to quickly reoptimize a slightly perturbed version of the original SDP, after the addition of cutting planes. We also describe a preliminary branch and bound approach. Computational results on a variety of maxcut problems are also discussed.

5.2 Directions for future work

We conclude the thesis with the following interesting avenues for future research.

1. It is important to investigate second order cone programming (SOCP) relaxations for combinatorial optimization problems. Recently Kim and Kojima [64] have investigated second order cone programming relaxations of nonconvex quadratic optimization problems. They show that the SOCP relaxation is a reasonable compromise between the effectiveness of the SDP relaxation, and the low computational cost of the LP relaxation. Again, there are a lot of problems (see Alizadeh and Goldfarb [2]) which can be directly cast as SOCP's rather than SDP's and hence more easily solved. It is also interesting that although the SOCP lies in between the LP and the SDP no approximation algorithms have been developed, which involve relaxing integer programs to SOCP's. The SOCP formulation of Muramatsu and Suzuki [88] (Section 4.5.2 in Chapter 4) is a step in this direction. It will also be interesting to consider SOCP relaxations of SDP's, which should better capture the smoothness inherent in the SDP problem. The interior point cutting plane scheme discussed in Chapter 3 can also solve an SOCP, although we envisage that these would longer be competitive with interior points methods for SOCP.

2. The spectral bundle scheme discussed in Chapter 2 is only a first order method. Recently Oustry [93] has come up with a second order scheme, but a practical implementation is still lacking. Moreover the spectral bundle method has a number of adjustable parameters; chief among these is the weight u which appears in the regularization term penalizing the displacement from the current iterate. Further investigation of choice of these parameters is necessary.

3. Another interesting question is whether there are practical simplex-like (active set) algorithms for $SOCP$ and the SDP. This has potential advantages in branch and cut approaches to solving integer programs in that once a problem has been solved, and subsequently changed in a small way, by adding an extra constraint, the dual simplex method can be used to reoptimize the new problem, starting from the original solution. It is also important to consider warm start strategies for branch and cut interior point SDP cutting plane schemes in this context (see Mitchell [81] for such a scheme with regard to the maxcut problem).

4. Burer et al [21, 23] have developed attractive heuristics for the max-cut and max stable set problems, based on imposing an additional rank two restriction on the maxcut SDP, and Lovasz theta SDP respectively (see section 4.5.1 in Chapter 4). The resulting problem is continuous but non-convex; the authors relate the minima of this problem to cuts and the stable sets in the graph, and have obtained excellent computational results on several large sized problems.

BIBLIOGRAPHY

[1] F. Alizadeh, *Interior Point Methods in Semidefinite Programming with applications to Combinatorial Optimization*, SIAM Journal on Optimization, 5 (1995), pp. 13-51.

[2] F. Alizadeh and D. Goldfarb, *Second Order Cone Programming*, RUTCOR RRR Report number 51-2001, Rutgers University, November 2001.

[3] F. Alizadeh, J.P.A. Haeberly and M.L. Overton, *Primal-dual interior-point methods for semidefinite programming : Convergence rates, stability and numerical results*, SIAM Journal on Optimization, 8(1998), pp. 746-768.

[4] F. Alizadeh, J.P.A. Haeberly and M.L. Overton, *Complementarity and Nondegeneracy in Semidefinite Programming*, Mathematical Programming, 77(1997), pp. 111-128.

[5] F. Alizadeh and S. Schmieta, *Symmetric Cones, Potential Reduction Methods and Word-By-Word Extensions*, In : H. Wolkowicz, R. Saigal, and L. Vandenberghe, eds., *Handbook of Semidefinite Programming*, Kluwer Academic Publishers, Boston-Dordrect-London, 2000, pp. 195-233.

[6] E.J. Anderson and P. Nash, *Linear Programming in Infinite-Dimensional Spaces*, Wiley, New York, 1987.

[7] M.F. Anjos and H. Wolkowicz, *Strengthened Semidefinite Relaxations via a Second Lifting for the Maxcut problem*, Discrete Applied Mathematics, 119(2002), pp. 79-106.

[8] E. Balas, S. Ceria and G. Cornuejols, *A lift and project cutting plane algorithm for mixed 0-1 programs*, Mathematical Programming, 58(1993), pp. 295-324.

[9] F. Barahona and R. Anbil, *The volume algorithm : producing primal solutions with a subgradient method*, Mathematical Programming, 87(2000), pp. 385-399.

[10] F. Barahona and A.R. Mahjoub, *On the cut polytope*, Mathematical Programming, 44(1989), pp. 157-173.

[11] G.P. Barker and D. Carlson, *Cones of diagonal dominant matrices*, Pacific Journal of Mathematics, 57(1975), pp. 15-32.

[12] S.J. Benson, Y. Ye and X. Zhang, *Solving Large-Scale Semidefinite Programs for Combinatorial Optimization*, SIAM Journal on Optimization, 10 (2000), pp. 443-461.

[13] A. Ben-Tal, A. Ben-Isreal, and E. Rosinger, *A Helly type theorem and semi-infinite programming*. In C.V. Coffman and G.J. Fix, editors, *Constructive Approaches to Mathematical models*, pages 127-135. Academic Press, New York 1979.

[14] A. Ben Tal and A. Nemirovskii, *Lectures on Modern Convex Optimization : Analysis, Algorithms, and Engineering Applications*, MPS-SIAM Series on Optimization, 2001.

[15] J. Berry and M. Goldberg, *Path optimization for graph partitioning problems*, Discrete Applied Mathematics, 90(1999), pp. 27-50.

[16] J.F. Bonnans and A. Shapiro, *Perturbation Analysis of Optimization Problems*, Springer-Verlag New York, 2001.

[17] B. Borchers, *SDPLIB 1.2, A Library of Semidefinite Programming Test Problems*, Optimization Methods and Software. 11(1999), pp. 683-690.

[18] J.M. Borwein, *Direct theorems in semi-infinite convex programming*, Mathematical Programming, 21 (1981), pp. 301-318.

[19] S. Boyd, L. El Ghaoui, E. Feron and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, SIAM Studies in Applied Mathematics, SIAM, Philadelphia, USA, 1994.

[20] S. Burer, R.D.C. Monteiro and Y. Zhang, *Solving a Class of Semidefinite Programs via Nonlinear Programming*, Technical Reports TR99-17 and 23 (combined), Dept. of Computational and Applied Mathematics, Rice University, October 2001.

[21] S. Burer, R.D.C. Monteiro and Y. Zhang, *Rank-Two Relaxation Heuristics for Max-Cut and Other Binary Quadratic Programs*, SIAM Journal on Optimization, 12(2001), pp. 503-521.

[22] S. Burer, R.D.C. Monteiro and Y. Zhang, *A Computational Study of a Gradient-Based Log-Barrier Algorithm for a Class of Large Scale SDP's*, Technical Report TR01-11, Dept. of Compuational and Applied Mathematics, Rice University, June 2001.

[23] S. Burer, R.D.C. Monteiro and Y. Zhang, *Maximum Stable Set Formulations and Heuristics Based on Continuous Optimization*, Technical Report TR00-34, Dept. of Computational and Applied Mathematics, Rice University, December 2000.

[24] A.R. Conn, N.I.M. Gould and P.L. Toint, *Trust-Region Methods*, MPS-SIAM Series on Optimization, 2000.

[25] ILOG CPLEX 6.5, *ILOG Inc.*, 1080 Linda Vista Avenue, Mountain View, CA 94043.

[26] J. Cullum, W.E. Donath and P. Wolfe, *The minimization of certain nondifferentiable sums of eigenvalues of symmetric matrices*, Mathematical Programming Study, 3(1975), pp. 35-55.

[27] L. El Ghaoui and S.I. Niculescu, *Advances in Linear Matrix Inequality Methods in Control*, SIAM, Philadelphia, 2000.

[28] E. de Klerk, *Aspects of Semidefinite Programming : Interior Point Algorithms and Selected Applications*, Applied Optimization Series, Volume 65, Kluwer Academic Publishers, May 2002.

[29] E. de Klerk, *Interior point methods for semidefinite programming*, Ph.D. thesis, ITS/TWI/SSOR, Techncial University of Delft, 1997.

[30] E. de Klerk and D. Paschenik, *Approximating the stability number of graph via copositive programming*, SIAM Journal on Optimization, 12(2002), pp. 875-892.

[31] W.E. Donath and A.J. Hoffman, *Lower bounds for the partitioning of graphs*, IBM Jl. of Research and Development 17(1973), pp. 120-125.

[32] A. Frieze and M.R. Jerrum, *Improved approximation algorithms for Max k-cut and Max Bisection*, Algorithmica 18, pp. 61-77.

[33] M. Fukuda, M. Kojima and M. Shida, *Lagrangian Dual Interior-Point Methods for Semidefinite Programs*, SIAM Journal on Optimization, 12(2002), pp. 1007-1031.

[34] M. Fukuda, M. Kojima, K. Murota and K. Nakata, *Exploiting Sparsity in Semidefinite Programming Via Matrix Completion I : General Framework*, SIAM Journal on Optimization, 11(2000), pp. 647-674.

[35] K. Glashoff and S.A. Gustafson, *Linear Optimization and Approximation*, Springer Verlag, New York, 1983.

[36] M. Goemans and D.P. Williamson, *Improved approximation algorithms for max cut and satisfiability problems using semidefinite programming*, J. A.C.M 42 (1995), pp. 1115-1145.

[37] J.L. Goffin and J.P. Vial, *Multiple Cuts in the Analytic Center Cutting Plane Method*, SIAM Journal on Optimization, 11 (2000), pp. 266-288.

[38] J.L. Goffin and J.P. Vial, *Convex nondifferentiable optimization : a survey focused on the analytic center cutting plane method*, Logilab Technical Report, Department of Management Studies, University of Geneva, Switzerland, February 1999.

[39] G.H. Golub and C.F. Van Loan, *Matrix Computations*, Third edition, The John Hopkins University Press, Baltimore, 1996.

[40] J. Gondzio, *Warm Start of the Primal-Dual Method Applied in the Cutting Plane Scheme*, Mathematical Programming, 83(1998), pp. 125-143.

[41] B. Grone, C.R. Johnson, E. Marques de Sa, and H. Wolkowicz, *Positive definite completions of partial Hermitian matrices*, Linear Algebra and its Applications, 58(1984), pp. 109-124.

[42] M. Grotschel, L. Lovasz and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer Verlag, Berlin, 1993.

[43] G. Gruber, *On Semidefinite Programming and Applications in Combinatorial Optimization*, Ph.D. thesis, Department of Mathematics, University of Klagenfurt, November 2000.

[44] J. Hastad, *Some optimal inapproximability results*, In Proc. of the 29th ACM Symposium on Theory and Computing, 1997.

[45] C. Helmberg, *Semidefinite Programming for Combinatorial Optimization*, ZIB-Report ZR-00-34, Konrad-Zuse-Zentrum Berlin, October 2000.

[46] C. Helmberg, *A Cutting Plane Algorithm for Large Scale Semidefinite Relaxations*, ZIB-Report ZR-01-26, Konrad-Zuse-Zentrum Berlin, October 2001.

[47] C. Helmberg, *Fixing Variables in Semidefinite Relaxations*, SIAM Journal on Matrix Analysis and Applications, 21(2000), pp. 952-969.

[48] C. Helmberg, *SBmethod : A C++ Implementation of the Spectral Bundle Method*, ZIB-Report ZR-00-35, Konrad -Zuse-Zentrum Berlin, October 2000.

[49] C. Helmberg, *Semidefinite Programming Homepage*, *http://www.mathematik.uni-kl.de/ helmberg/semidef.html*

[50] C. Helmberg and K.C. Kiwiel, *A Spectral Bundle Method with bounds*, ZIB Preprint SC-99-37, Konrad-Zuse-Zentrum Berlin, December 1999.

[51] C. Helmberg and F. Oustry, *Bundle methods to minimize the maximum eigenvalue function.* In : H. Wolkowicz, R. Saigal, and L. Vandenberghe, eds., *Handbook of Semidefinite Programming*, Kluwer Academic Publishers, Boston-Dordrect-London, 2000, pp. 307-337.

[52] C. Helmberg and F. Rendl, *A Spectral Bundle Method for Semidefinite Programming*, SIAM Journal on Optimization, 10 (2000), pp. 673-696.

[53] C. Helmberg and F. Rendl, *Solving Quadratic (0,1) Problems by Semidefinite Programs and Cutting Planes*, Mathematical Programming, 82(1998), pp. 291-315.

[54] C. Helmberg, F. Rendl, R. Vanderbei and H. Wolkowicz, *An interior point method for semidefinite programming*, SIAM Journal on Optimization, 6(1996), pp. 673-696.

[55] R. Hettich and K.O. Kortanek, *Semi-Infinite Programming : Theory, Methods, and Applications*, SIAM Review 35(3) (1993), pp. 380-429.

[56] J.B. Hiriart Urruty and C. Lemarechal, *Convex Analysis and Minimization Algorithms I*, Springer Verlag, Berlin, Heidelberg, 1993.

[57] J.B. Hiriart Urruty and C. Lemarechal, *Convex Analysis and Minimization Algorithms II*, Springer Verlag, Berlin Heidelberg, 1993.

[58] R.A. Horn and C.R. Johnson, *Matrix Analysis*, Cambridge University Press, New York, 1990.

[59] R.A. Horn and C.R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, New York, 1991.

[60] G. Iyengar and M.T. Cezik, *Cutting planes for mixed 0-1 semidefinite programming*, 8th International Conference on Integer Programming and Combinatorial Optimization (IPCO), Utrecht, Netherlands, 2001.

[61] H. Karloff, *How good is the Goemans-Williamson MAX CUT algorithm?*, SIAM Journal on Computing, 29(1999), pp. 336-350.

[62] N.K. Karmarkar, *A new polynomial time algorithm for linear programming*, Combinatorica, 4(1984), pp. 373-395.

[63] R.M. Karp, *Reducibility among combinatorial problems*, In R.E. Miller and J.W. Thatcher, editors, Complexity of Computer Computation, pp. 85-103, Plenum Press, New York, 1972.

[64] S. Kim and M. Kojima, *Second order cone programming relaxations of nonconvex quadratic optimization problems*, Optimization Methods and Software, 15(2001), pp. 201-224.

[65] K.C. Kiwiel, *An aggregate subgradient method for nonsmooth convex minimization*, Mathematical Programming 27(1983), pp. 320-341.

[66] K.C. Kiwiel, *Methods of Descent for Nondifferentiable Optimization*, Lecture Notes in Mathematics, vol. 1133, Springer, Berlin, Heidelberg (1985).

[67] K.C. Kiwiel, *Proximity Control in bundle methods for convex nondifferentiable minimization*, Mathematical Programming 46 (1990), pp. 105-122.

[68] M. Kojima, S. Shindoh and S. Hara, *Interior-point methods for the monotone linear complementarity problem in symmetric matrices*, SIAM Journal on Optimization, 7(1997), pp. 86-125.

[69] K. Krishnan and J.E. Mitchell, *Linear programming approaches to semidefinite programming problems*, Technical Report, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, May 2001.

[70] K. Krishnan and J.E. Mitchell, *Semi-infinite linear programming approaches to semidefinite programming problems*, Technical Report, Dept. of Mathematical Sciences, Rensselaer Polytechnic Institute, August 2001 (to appear in a special issue of *Journal of Combinatorial Optimization*).

[71] K. Krishnan and J.E. Mitchell, *An SDP approach for the maxcut problem*, Technical Report, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, June 2002.

[72] J.B. Lasserre, *Global optimization with polynomials and the problem of moments*, SIAM Journal on Optimization, 11(2001), pp. 796-817.

[73] M. Laurent, *A tour d'horizon on positive semidefinite and Euclidean distance matrix completion problems*, In Topics in Semidefinite and Interior Point Methods, Volume 18 of The Fields Institute for Research in Mathematical Sciences, Communications Series, Providence, Rhode Island, 1998, AMS.

[74] M. Laurent and S. Poljak, *On a positive semidefinite relaxation of the cut polytope*, Linear Algebra and its Applications, 223(1995), pp. 439-461.

[75] C. Lemarechal, *Nondifferentiable Optimization*. In : G.L. Nemhauser, A.H.G. Rinnooy Kan, and M.J. Todd, eds., *Optimization*, North-Holland, Amsterdam-New York-Oxford-Tokyo, 1989, pp. 529-572.

[76] A.S. Lewis and M.L. Overton, *Eigenvalue Optimization*, Acta Numerica 5(1996), pp. 149-190.

[77] L.S. Lobo, L. Vandenberghe, S. Boyd and H. Lebret, *Second Order Cone Programming*, Linear Algebra and its Applications, 284(1998), pp. 193-228.

[78] L. Lovasz, *On the Shannon capacity of a graph*, IEEE Transactions on Information Theory, 25(1979), pp. 1-7.

[79] L. Lovasz and A. Schrijver, *Cones of matrices and set functions and 0-1 optimization*, SIAM Journal on Optimization, 1(1991), pp. 166-190.

[80] J.E. Mitchell and B. Borchers, *Solving real-world linear ordering problems using a primal-dual interior point cutting plane method*, Annals of Operations Research, 62(1996), pp. 253-276.

[81] J.E. Mitchell, *Restarting after branching in the SDP approach to MAX-CUT and similar combinatorial optimization problems*, Journal of Combinatorial Optimization, 5(2001), pp. 151-166.

[82] J.E. Mitchell, *Computational Experience with an Interior Point Cutting Plane Algorithm*, SIAM Journal on Optimization, 10(2000), pp. 1212-1227.

[83] J.E. Mitchell, *Polynomial interior point cutting plane methods*, Technical Report, Dept. of Mathematical Sciences, Rensselaer Polytechnic Institute, July 2001.

[84] J.E. Mitchell and S. Ramaswamy, *A Long-Step, Cutting Plane Algorithm for Linear and Convex Programming*, Annals of OR, 99(2000), pp. 95-122.

[85] J.E. Mitchell and M.J. Todd, *Solving combinatorial optimization problems using Karmarkar's algorithm*, Mathematical Programming, 56(1992), pp. 245-284.

[86] H. Mittlemann, *Decision Tree for Optimization Software*, *http://plato.la.asu.edu/guide.html*

[87] R.D.C Monteiro, *Primal Dual path following algorithms for semidefinite programming*, SIAM Journal on Optimization, 7(1997), pp. 663-678.

[88] M. Muramatsu and T. Suzuki, *A New Second Order Cone Programming Relaxation for MAX-CUT Problems*, Technical Report CS-01-01, Dept. of Computer Science, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu-shi, Tokyo, Japan, May 2001.

[89] Y.E. Nesterov, *Semidefinite relaxation and nonconvex quadratic optimization*, Optimization Methods and Software, 9(1998), pp. 141-160.

[90] Y. Nesterov and A. Nemirovskii, *Interior Point Polynomial Algorithms in Convex Programming*, in SIAM Studies in Applied Mathematics, Philadelphia, PA, 1993.

[91] Optimization Online, *http://www.optimization-online.org/*

[92] M. Oskoorouchi, *The Analytic Center Cutting Plane Method with Semidefinite Cuts*, Ph.D. thesis, Faculty of Management, McGill University, Montreal, Canada, May 2002.

[93] F. Oustry, *A second order bundle method to minimize the maximum eigenvalue function*, Mathematical Programming 89 (2000), pp. 1-33.

[94] F. Oustry and C. Lemarechal, *Semidefinite relaxations and Lagrangian duality with applications to combinatorial optimization*, RR-3170, INRIA Rhone-Alpes, June 1999.

[95] M.L. Overton, *Large-Scale Optimization of Eigenvalues*, SIAM Journal on Optimization, 2(1992), pp. 88-120.

[96] C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization : Algorithms and Complexity*, Prentice Hall, 1982.

[97] B. Parlett *Personal Communication*, May 2002.

[98] B. Parlett and K. Krishnan, *Diagonal perturbations of indefinite matrices to restore positive semidefiniteness*, in preparation.

[99] G. Pataki, *On the Rank of Extreme Matrices in Semidefinite Programs and the Multiplicity of Optimal Eigenvalues*, Mathematics of Operations Research 23(2), pp. 339-358.

[100] G. Pataki, *The Geometry of Semidefinite Programming*, In : H. Wolkowicz, R. Saigal, and L. Vandenberghe, eds., *Handbook of Semidefinite Programming*, Kluwer Academic Publishers, Boston-Dordrect-London, 2000, pp. 29-65.

[101] G. Pataki, *Cone-LP's and semidefinite programs : geometry and simplex type method*, In Lecture Notes in Computer Science, 1084(1996), pp. 162-174.

[102] G. Pataki and S. Schmieta, *The DIMACS library of semidefinite -quadratic-linear programs*, Computational Optimization Research, Columbia University, NY, November 1999.

[103] S. Poljak and Z. Tuza, *The maxcut problem - a survey*, In Special Year on Combinatorial Optimization, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, AMS, 1995.

[104] J. Renegar, *A Mathematical View of Interior-Point Methods in Convex Optimization*, MPS-SIAM Series on Optimization, 2001.

[105] R.T. Rockafellar, *Convex Analysis*, Princeton University Press, 1970.

[106] S. Schmieta and F. Alizadeh, *Associative and Jordan Algebras, and Polynomial Time Interior Point Algorithms for Symmetric Cones*, Mathematics of Operations Research, 26(2001), pp. 543-564.

[107] H. Schramm and J. Zowe, *A version of the bundle idea for minimizing a nonsmooth function : Conceptual idea, convergence analysis, numerical results*, SIAM Journal on Optimization 2 (1992), pp. 121-152.

[108] H.D. Sherali and W.P. Adams, *A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems*, SIAM Journal on Discrete Mathematics, 3(1990), pp. 411-430.

[109] N.Z. Shor, *Quadratic Optimization Problems*, Soviet Journal of Circuits and System Sciences, 25(1987), pp. 1-11.

[110] M.J. Todd, *Semidefinite Optimization*, Acta Numerica 10 (2001), pp. 515-560.

[111] M.J. Todd, *A study of search directions in interior point methods for semidefinite programming*, Optimization Methods and Software, 12(1999) pp. 1-46.

[112] M.J. Todd, K.C. Toh and R.H. Tutuncu, *On the Nesterov-Todd direction in semidefinite programming*, SIAM Journal on Optimization, 8(1998), pp. 769-796.

[113] K.C. Toh and M. Kojima, *Solving some large scale semidefinite programs via the conjugate residual method*, SIAM Journal on Optimization, 12(2002), pp. 669-691.

[114] K.C. Toh, M.J. Todd and R.H. Tutuncu, *SDPT3 - A Matlab Software Package for Semidefinite Programming. Version 2.1*, Optimization Methods and Software, 11(1999), pp. 1-49.

[115] L. Vandenberghe and S. Boyd, *Semidefinite Programming*, SIAM Review, 38(1996), pp. 49-95.

[116] R. Vanderbei, *LOQO : An interior point code for quadratic programming*, Optimization Methods and Software, 12(1999), pp. 451-484.

[117] R. Vanderbei and H. Benson, *On Formulating Semidefinite Programming Problems as Smooth Convex Nonlinear Optimization Problems*, ORFE-99-01, Princeton University, NJ, January 2000.

[118] H. Wolkowicz, R. Saigal and L. Vandenberghe, *Handbook on Semidefinite Programming*, Kluwer Academic Publishers, 2000.

[119] S. Wright, *Interior-Point Methods Online*, *http://www-unix.mcs.anl.gov/otc/InteriorPoint/*

[120] Y. Ye, *A .699 Approximation Algorithm for Max Bisection*, Mathematical Programming, 90(1) (2001), pp. 101-111.

[121] Y. Zhang, *Solving large-scale linear programs by interior point methods under the MATLAB environment*, Optimization Methods and Software, 10(1998), pp. 1-31.

[122] Y. Zhang, *On extending some primal-dual interior point algorithms from linear programming to semidefinite programming*, SIAM Journal on Optimization, 8(1998), pp. 365-386.

[123] U. Zwick, *Outward rotations : a tool for rounding solutions of semidefinite programming relaxations, with applications to MAXCUT and other problems*, In Proc. of 31st STOC, 1999.

APPENDIX A
Glossary of symbols and notation

k	Number of constraints in the SDP
n	size of the symmetric matrices in SDP
r	usually the rank of a symmetric matrix
$\mathcal{R}$	real numbers
$\mathcal{Z}$	integers
$\mathcal{R}^n$	real column vector of dimension n
$\mathcal{R}_+{}^n$	nonnegative real column vector
$\mathcal{R}_{++}{}^n$	strictly positive real column vector
$\mathcal{M}^n(\mathcal{R}^{n\times n})$	real $n \times n$ matrices
$\mathcal{S}^n$	real symmetric matrices of size n
$\mathcal{S}_+{}^n$	real positive semidefinite (psd) matrices of size n
$\mathcal{S}_{++}{}^n$	real positive definite (pd) matrices of size n
$\mathcal{L}^n$	lower triangular matrix of size n
$\mathcal{L}_+{}^n$	lower triangular matrix with nonnegative diagonal entries
$\mathcal{L}_{++}{}^n$	lower triangular matrix with strictly positive diagonal entries
$I(I_n)$	identity matrix of appropriate size (size n)
e	vector of all ones of appropriate size
e_i	ith column of I
$\lambda_i(A)$	ith eigenvalue of A
$\lambda_{min}(A)(\lambda_{max}(A))$	minimal (maximal) eigenvalue of A
Λ_A	diagonal matrix in the spectral decomposition $A = P\Lambda_A P^T$
$\det(A)$	determinant of A
$\text{rank}(A)$	rank of A
A^T	transpose of A

$\mathrm{tr}(A)(\mathrm{Trace}(A))$	trace of A
$A \bullet B$	inner product of A and B in $\mathcal{S}^n$
$\|\|A\|\|(\|\|A\|\|_F)$	Frobenius norm of A, $\|\|A\|\|_F = \sqrt{A \bullet A}$
$A(:, i)$	ith column of matrix A
$A(:, i:j)$	sub-matrix obtained by taking columns i to j of A
$A(i:j, k:l)$	sub-matrix obtained by taking rows i to j, and columns k to l of A
$X_{\Sigma,\Sigma}$	sub-matrix obtained from the entries in the discrete set Σ
$\mathrm{Diag}(v)$	diagonal matrix with components of v along main diagonal
$\mathrm{diag}(A)$	the diagonal of A as a column vector
$\mathrm{conv}(S)$	the convex hull of a set S
$G = (V, E)$	graph with vertex set V and edge set E
V	set of nodes, usually $V = \{1, \ldots, n\}$
E	set of edges, $E \subset \{\{i, j\} : 1 \leq i < j \leq n\}$
$\{i, j\}$	edge with end points i and j
$\delta(S)$	the set of edges with exactly one end point in S (cut)

APPENDIX B
The link between spectral bundle and the cutting plane LP approach

We discussed the spectral bundle approach in Chapter 2, and a cutting plane LP approach, based on a semi-infinite formulation of (SDD), in Chapter 3. Despite the differences in the way we motivated these two approaches, the two are in fact very similar. We demonstrate this close similarity here. Other good references include Lemarechal [75], Helmberg [45], and Helmberg and Rendl [52].

We can rewrite (SDD) as the following eigenvalue optimization problem

$$\begin{array}{rcl} \max_y \quad f(y) & & \text{where} \\ f(y) & = & a\lambda_{min}(C - \mathcal{A}^T y) + b^T y \end{array} \tag{B.1}$$

To see this consider the following. If we add the redundant constraint $\text{Trace}(X) = a$ in (SDP), the new (SDD) is then

$$\begin{array}{lrcll} \max_y b^T y + av & & & & \\ \text{s.t.} & S & = & C - \mathcal{A}^T y - vI & (SDD) \\ & S & \succeq & 0 & \end{array}$$

Here v is the dual variable corresponding to $\text{Trace}(X) = a$. The constraint $S \succeq 0 \Leftrightarrow \lambda_{min}(S) \geq 0$. However the $\text{Trace}(X) = a$ constraint forces $\lambda_{min}(S) = 0$. Thus we have

$$\begin{array}{rcl} \lambda_{min}(S) = 0 & \Rightarrow & \lambda_{min}(C - \mathcal{A}^T y - vI) = 0 \\ & \Rightarrow & v = \lambda_{min}(C - \mathcal{A}^T y) \end{array}$$

Substituting this value of v in the objective function of (SDD) gives (B.1).

Now assume that we have a set of points $y = y^1, \ldots, y^m$, and we know the function values $f(y^i)$, $i = 1, \ldots, m$, and subgradients $(b - \mathcal{A}(p_i p_i^T))$, $i = 1, \ldots, m$ (where p_i is a normalized eigenvector corresponding to $\lambda_{min}(C - \mathcal{A}^T y^i)$) at these points. To simplify the notation, and without loss of generality, we will assume that

$a = 1$ in the resulting discussion.

We can construct the following overestimate $\hat{f}_m(y)$ for $f(y)$.

$$\begin{aligned} \hat{f}_m(y) &= \min_{i=1,\ldots,m} p_i p_i^T \bullet (C - \mathcal{A}^T y) + b^T y \\ &\geq f(y) \end{aligned}$$

To see this note that from the subgradient inequality for a concave function we have

$$\begin{aligned} \lambda_{min}(C - \mathcal{A}^T y) &\leq \lambda_{min}(C - \mathcal{A}^T y^i)) + p_i p_i^T \bullet (C - \mathcal{A}^T y - (C - \mathcal{A}^T y^i)) \\ &= p_i p_i^T \bullet (C - \mathcal{A}^T y), \quad i = 1, \ldots, m \end{aligned}$$

where the last equality follows from the fact that $\lambda_{min}(C - \mathcal{A}^T y^i) = p_i p_i^T \bullet (C - \mathcal{A}^T y^i)$. We now maximize this overestimate instead, i.e.

$$\max_y \hat{f}_m(y) = \max_y b^T y + \min_{i=1,\ldots,m} \{p_i p_i^T \bullet (C - \mathcal{A}^T y)\}$$

which can be recast as the following linear program

$$\begin{array}{lrcl} \max & b^T y + v & & \\ \text{s.t.} & v & \leq & p_i p_i^T \bullet (C - \mathcal{A}^T y), \quad i = 1, \ldots, m \end{array} \tag{B.2}$$

This is exactly the problem obtained by considering a discretization of (SDD) (as described in Chapter 2).

Before, we begin our discussion of the spectral bundle approach, let us first consider the proximal bundle approach (see Kiwiel [67], and Lemarechal [75]). This approach considers a polyhedral approximation of the minimum eigenvalue function like our cutting plane LP approach; however unlike our LP approach we do not maximize the overestimate $\hat{f}_m(y)$, but rather $\hat{f}_m(y) - \frac{1}{2}u|y - y^m|^2$ (for some chosen $u > 0$). This gives a QP with a polyhedral feasible region. To simplify our notation, and without loss of generality, we assume that y^m is the iterate where $f(y)$ is a maximum, i.e. $f(y^m) \geq f(y^i)$, $i = 1, \ldots, m-1$. Let us also introduce some notation

here. Let

$$\begin{aligned} q_i &= b^T(y^m - y^i) + (\lambda_{min}(C - \mathcal{A}^T y^m) - \lambda_{min}(C - \mathcal{A}^T y^i)) - \\ &\qquad (b - \mathcal{A}(p_i p_i^T))^T (y^m - y^i) \\ d &= y - y^m \end{aligned}$$

Here q_i refers to the linearization error due to the subgradient $(b - \mathcal{A}(p_i p_i^T))$ (computed at y^i) at the curent iterate y^m. Since we are dealing with a concave function $q_i \leq 0$, $i = 1, \ldots, m-1$. Using the above notation, we can write $\hat{f}^m(y)$ as follows :

$$\begin{aligned} \hat{f}^m(y) &= f(y^m) + \min\{-q_i + (b - \mathcal{A}(p_i p_i^T))^T d, i = 1, \ldots, m\} \\ &= b^T y + \min\{-q_i + \lambda_{min}(C - \mathcal{A}^T y^m) - \mathcal{A}(p_i p_i)^T d, i = 1, \ldots, m\} \end{aligned} \tag{B.3}$$

with $q_m = 0$. The final problem we solve to obtain the search direction d is then

$$\begin{aligned} &\max \quad b^T y + v - \tfrac{1}{2} u d^T d \\ &\text{s.t.} \qquad\qquad v \leq -q_i + \lambda_{min}(C - \mathcal{A}^T y^m) - \mathcal{A}(p_i p_i^T) d, \quad i = 1, \ldots, m \end{aligned} \tag{B.4}$$

If we set $u = 0$ in (B.4) we identically have (B.2). The dual to this subproblem is (B.5). Due to strong duality the two problems (B.4) and (B.5) are equivalent. Let us write out the Lagangian dual to (B.4)

$$\min_{\lambda \geq 0} \max_{d,v} b^T d + v - \frac{1}{2} u d^T d + \sum_{i=1}^{m} \lambda_i (-q_i - \mathcal{A}(p_i p_i^T))^T d - v) + \lambda_{min}(C - \mathcal{A}^T y^m) + b^T y^m$$

An approximation to this problem is (B.5).

$$\begin{aligned} \min_{\lambda \geq 0, \sum_{i=1}^m \lambda_i = 1, \sum_{i=1}^{m-1} \lambda_i q_i \leq \epsilon} \quad & \tfrac{1}{2u} || b - \mathcal{A}(\textstyle\sum_{i=1}^m \lambda_i p_i p_i^T) ||^2 + \\ & b^T y^m + \lambda_{min}(C - \mathcal{A}^T y^m) \end{aligned} \tag{B.5}$$

This is obtained by noting that the term $(1 - \sum_{i=1}^m \lambda_i) v$ attains its maximum when $\sum_{i=1}^m \lambda_i = 1$; besides $(b - \mathcal{A}(\sum_{i=1}^m \lambda_i p_i p_i^T))^T d - \frac{1}{2} u d^T d$ is a strictly concave function, and attains its maximum when

$$d^m = \tfrac{1}{u}(b - \mathcal{A}(\textstyle\sum_{i=1}^m \lambda_i p_i p_i^T)) \tag{B.6}$$

The constraint $\sum_{i=1}^{m-1} \lambda_i q_i \leq \epsilon$ suggests that we include only the subgradients $(b - \mathcal{A}(p_i p_i^T))$, $i = 1, \ldots, m-1$ from the previous iterations in the current polyhedral approximation of the ϵ subdifferential, whose linearization errors q_i, $i = 1, \ldots, m-1$ are small. For more details on this polyhedral approximation see Lemarechal [75]. Carry out the updates.

$$\begin{aligned} \bar{W}^m &= \sum_{i=1}^{m} \lambda_i p_i p_i^T \\ y^{m+1} &= y^m + d^m \end{aligned}$$

We then compute another ϵ subgradient $p_{m+1}{p_{m+1}}^T$, where p_{m+1} is an normalized eigenvector corresponding to $\lambda_{min}(C - \mathcal{A}^T y^{m+1})$. Now let

$$W^{m+1} = \text{conv}\{p_i p_i^T, i = 1, \ldots, m+1\} \tag{B.7}$$

This ensures that $\bar{W}^m$, and $p_{m+1}{p_{m+1}}^T \in \mathcal{W}^{m+1}$; the two conditions that are required to guarantee convergence of the scheme (see Lemarechal [75], Kiwiel [67], and Helmberg [51]). The term $\bar{W}^m$ plays the role of an aggregate subgradient here, and is used to keep the number of variables λ in the problem (B.5) small.

The spectral bundle method on the other hand utilizes a semidefinite cutting plane model for maximizing the minimum eigenvalue function. This follows from the alternative expression (2.24) (see section 2.6) for the subdifferential of the minimum eigenvalue function. Based on this expression, let us define

$$\mathcal{W}^{m+1} = \{U = P_{m+1} V {P_{m+1}}^T : V \in {\mathcal{S}_+}^{r^{m+1}}, \text{Trace}(V) = 1\} \tag{B.8}$$

Here r^{m+1} is the multiplicity of $\lambda_{min}(S^{m+1})$. Also $P_{m+1} \in \mathcal{R}^{n \times r_{m+1}}$ is an orthonormal basis for the corresponding eigenspace. This set (B.8) is no longer a polyhedron unlike (B.7), but actually an ellipsoid (also see the discussion in section 1.3.2). Thus, we would need an infinite number of terms in the convex hull representation of W^{m+1} to capture $\mathcal{W}^{m+1}$. In fact we always have

$$W^i \subset \mathcal{W}^i, \;\; i = 1, \ldots, m+1$$

We can incorporate an aggregate subgradient matrix $\bar{W}^{m+1}$, as in the proximal

bundle scheme, to try and keep the number of columns r^{m+1} in P small. Helmberg and Rendl [52] consider the following subset for $\mathcal{W}^i$ in every iteration.

$$\mathcal{W}^i \;=\; \{\alpha \bar{W}^i + P_i V P_i^T : \alpha + \text{Trace}(V) = 1, \alpha \geq 0, V \in {\mathcal{S}_+}^{r^i}\}$$

In case, we do not have $\bar{W}^i$, then we would need at least $O(\sqrt{k})$ columns in P_i. This is because the multiplicity r^i of $\lambda_{min}(C - \mathcal{A}^T y^i)$, $i = 1, \ldots, m$ goes up steadily, as the optimization proceeds, since the smaller eigenvalues tend to coalesce together. An upper bound on r^i is given by Theorem 16 (see section 2.5), i.e. $r^i \leq r$, where

$$\tfrac{r(r+1)}{2} \;\leq\; k+1 \tag{B.9}$$

Here $k+1$ is the number of constraints in (SDP), including the redundant constraint $\text{tr}(X) = 1$. Thus, the quadratic subproblem $(QSDP_{sb})$ the spectral bundle method solves in the $(m+1)$th iteration is given by :

$$\begin{array}{lrcll} \min & \frac{1}{2u}||b - \mathcal{A}(X)||^2 & & & \\ \text{s.t.} & X & = & \alpha \bar{W}^m + P_{m+1} V {P_{m+1}}^T & (QSDP_{sb}) \\ & \alpha + \text{trace}(V) & = & 1 & \\ & \alpha & \geq & 0 & \\ & V & \succeq & 0 & \end{array}$$

where $V \in {\mathcal{S}_+}^{r^{m+1}}$. We solve $(QSDP_{sb})$ to obtain X, and then obtain d using (B.6), with X instead of $\sum_{i=1}^m \lambda_i p_i p_i^T$.

We conclude with the following observations :

1. Thw two essential differences between our LP approach, and the spectral bundle scheme are :

   - The spectral bundle scheme uses a non-polyhedral cutting plane model that approximates the minimum eigenvalue function. On the other hand, we consider a polyhedral cutting plane model. This explains the poor convergence of the LP cutting plane approaches we considered in Chapter 3.

- The spectral bundle scheme like its proximal bundle predecessor employs a quadratic regularization term, to penalize any displacement from the current iterate. The idea being : minimizing $\hat{f}_m$ (see B.3) to find the next iterate y^{m+1} makes sense only if $\hat{f}_m \approx f$ near y^m. This also ensures the solution to (B.5) is bounded. On the other hand we need additional box constraints to ensure that the feasible region to (B.2) is bounded, especially at the start of the cutting plane procedure.

2. The idea is to keep r (the number of columns in P) in ($QSDP_{sb}$) small so that this subproblem can be solved quickly. In the absence of the aggregate matrix $\bar{W}$ we will need as many columns in P as the multiplicity of $\lambda_{min}(S)$ at optimality.

3. The efficiency of the spectral bundle approach (for that matter the proximal bundle scheme) is in choosing a good u. As Lemarechal [75] remarks, this is an art in itself. The choice of a good u is what makes the spectral bundle approach of Chapter 2 superior to the cutting plane LP approach of Chapter 3.

4. This link between the two approaches, also motivates the spectral bundle as a cutting plane approach for the SDP. As mentioned earlier, the spectral bundle scheme actually provides a non-polyhedral cutting plane model to solving the underlying SDP.