Notation

$\mathbb{R}$ - Set of real numbers
$\mathbb{R}_+$ - Set of non-negative real numbers
$\mathbb{R}^n$ - Set of n-dimensional real vectors
$\mathbb{R}^n_+$ - Set of n-dimensional real vectors with non-negative entries

S^n - Unit sphere in $\mathbb{R}^n$

$\mathbb{S}^n$ - Set of symmetric $n \times n$ matrices
$\mathbb{S}^n_+$ - Set of symmetric positive semidefinite $n \times n$ matrices
$\mathbb{S}^n_{++}$ - Set of symmetric positive definite $n \times n$ matrices

$M \succ 0$ - M is symmetric and positive definite
$M \succeq 0$ - M is symmetric and positive semidefinite
$C \bullet X$ - Matrix inner product $= Tr(C^T X)$

PSD - Positive Semidefinite

QCQP - Quadratically Constrained Quadratic Program
SDP - Semidefinite program
SOCP - Second Order Cone Program
LP - Linear Program

Chapter 1

Introduction

In this thesis we will study quadratically constrained quadratic programs (QCQP)

Definition 1.1. The Quadratically Constrained Quadratic Program (QCQP) in the standard form is

$$\begin{array}{ll} \text{minimize} & x^T P_0 x + q_0^T x + r_0 \\ \text{subject to} & x^T P_k x + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m), \end{array} \tag{1.2}$$

where $x \in \mathbb{R}^n$ is a variable, and symmetric $n \times n$ matrices $P_0, P_1, \ldots, P_m \in S^n$, vectors $q_0, \ldots, q_m \in \mathbb{R}^n$ and scalars $r_1, \ldots, r_m \in \mathbb{R}$ are given.

The matrices $P_0, P_1, \ldots P_m$ are not necessarily positive semidefinite. Therefore the objective function as well as the constraints may be nonconvex. Problem (1.2) has been proved to be NP-hard in general [24], while several special subclasses of QCQP have been identified to be polynomially solvable (see [2]).

For example, the 0-1 constraint $x_i \in \{0, 1\}$ can be reformulated as $x^T e_i e_i^T x - e_i^T x = 0$. Thus QCQP includes 0-1 programming, which describes various NP-hard problems, such as knapsack, stable set, max cut etc. On the other hand, the above examples suggest that QCQP has many applications and is worth solving.

One of the possible approaches is relaxing QCQP to obtain convex problems which can be solved in polynomial time, namely linear programming (LP), second order cone programming (SOCP), or semidefinite programming (SDP).

Chapter 2

Conic optimization classes

In this section we will introduce basic optimization classes mentioned above, in particular the linear programming (LP), second order cone programming (SOCP), or semidefinite programming (SDP). We will state the problems in standard forms and their duals. The dual problems will be only mentioned here and will be derived in the next section. We will also show that LP is subclass of convex QCQP, convex QCQP is subclass of SOCP, and SOCP is subclass of SDP, i.e.

$$LP \subset \text{convex } QCQP \subset SOCP \subset SDP \subset QCQP. \tag{2.1}$$

2.1 Linear programming

When both, the objective and the constraint functions are linear (affine), the problem is called a linear program and it belongs to the Linear Programming class, or shortly LP. In this section we will introduce the standard form of LP and its dual. For reference and more information about this topic see i.e. [5] .

Definition 2.2. The primal–dual pair of linear programs in standard form is

$$\begin{array}{llcll} & \textit{Primal} & & & \textit{Dual} \\ \text{minimize} & c^T x, & & \text{maximize} & b^T y, \\ \text{subject to} & Ax = b, & & \text{subject to} & A^T y + s = c, \\ & x \in \mathbb{R}^n_+, & & & s \in \mathbb{R}^n_+, \end{array} \tag{2.3}$$

where $x \in R^n$, $y \in \mathbb{R}^m$, $s \in \mathbb{R}^n$ are the variables; the real $m \times n$ matrix A and vectors $b \in \mathbb{R}^m, c \in \mathbb{R}^n$ are given problem data.

Remark 2.4. Linear programs can be formulated in various forms (including $\geq, \leq$ inequalities, free variables, possibly some linear fractions in objective) but all of them can be transformed to the standard form.

2.2 Second order cone programing

The second order cone programming (SOCP) is a convex optimization class which can be solved with great efficiency using interior point methods. In this section we will introduce the standard form of SOCP and its dual. For reference and more information about this topic see [5] .

Let us first define second order cone.

Definition 2.5 (Second order cone)**.** We say Q_n is second order cone of dimension n if

$$Q_n = \{x \in \mathbb{R}^n \mid x = (x_0, \bar{x}) \in \mathbb{R} \times \mathbb{R}^{n-1}, \|\bar{x}\|_2 \leq x_0\}. \tag{2.6}$$

Definition 2.7 (SOCP)**.** The primal–dual pair of the Second Order Cone Program (SOCP) in the standard form is

$$\begin{array}{llcll} & \textit{Primal} & & & \textit{Dual} \\ \text{minimize} & c^T x, & & \text{maximize} & b^T y, \\ \text{subject to} & Ax = b, & & \text{subject to} & A^T y + s = c, \\ & x \in Q_n, & & & s \in Q_n, \end{array} \tag{2.8}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$ and $s \in \mathbb{R}^n$ are the variables; and $m \times n$ real matrix A, vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and second order cone Q_n are given problem data.

Remark 2.9. Second order cone programs can be formulated in various forms (including quadratic objective or several second order cone constraints of the affine functions), but all of them can be transformed to the standard form.

Remark 2.10. In general, any program of the form

$$\begin{array}{llcll} & \textit{Primal} & & & \textit{Dual} \\ \min & c^{1T} x^1 + \cdots + c^{kT} x^k, & & \max & b^T y, \\ \text{s.t.} & A^1 x^1 + \cdots + A^k x^k = b, & & \text{s.t.} & A^{iT} y + s^i = c^i, \\ & x^i \in Q_{n_i},\ (i = 1, \ldots, k), & & & s^i \in Q_{n_i},\ (i = 1, \ldots, k), \end{array} \tag{2.11}$$

is considered to be SOCP. The second order cone constraints can be also formulated as, $x = (x^1, \ldots, x^k) \in Q$, where Q is Cartesian product of second order cones,

$$Q = Q_{n_1} \times Q_{n_2} \times \cdots \times Q_{n_k}, \tag{2.12}$$

Since, such Q has all important properties of second order cone (see appendix), and algorithmic aspects of solving standard SOCP also work for this more general case [7,8]. In order to keep things simple, we will sometimes consider only the standard form stated in the Definition 2.7, but all the details can be also done for this more general form.

2.2.1 Relation to previous classes

Second Order Cone Programming includes convex LP as special case. We will show that SOCP in fact includes convex QP as a subclass. We will demonstrate procedure proposed in [4] used to reformulate convex QP as SOCP.

Let us have convex QCQP. In other words, suppose that $n \times n$ matrices A_k, $k = 0, \ldots, m$ are positive semidefinite.

$$\begin{array}{ll} \text{minimize} & x^T P_0 x + q_0^T x + r_0, \\ \text{subject to} & x^T P_k x + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m), \end{array} \tag{2.13}$$

First of all, rewrite problem equivalently as

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & x^T P_0 x + q_0^T x + r_0 \leq t, \\ & x^T P_k x + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m). \end{array} \tag{2.14}$$

To avoid tedious notation, without loss of generality, suppose that considered program already has linear objective function (i.e. $P_0 = 0$). Also suppose that we have separated all the linear constraints (ones where $P_k = 0$) and arrange them into more compact form $Ax = b$. Even if we did not, the following procedure will still be correct, but will result in more complicated formulation of linear constraints.

Each convex quadratic constraint

$$x^T P x + q^T x + r \leq 0 \tag{2.15}$$

can be transformed into the second order cone constraint. Suppose that $P \neq 0$ and rank $P = h$. Then there exists $n \times h$ matrix L such that $P = LL^T$. Such L can be computed by Choelsky factorization of P. Now rewrite (2.15) as

$$(L^T x)^T (L^T x) \leq -q^T x - r. \tag{2.16}$$

It can be easily verified that $w \in \mathbb{R}^t$, $\xi \in \mathbb{R}$ and $\eta \in \mathbb{R}$ satisfy

$$w^T w \leq \xi\eta, \quad \xi \geq 0 \ \text{ and } \ \eta \geq 0$$

if and only if they satisfy

$$\left\| \begin{pmatrix} \xi - \eta \\ 2w \end{pmatrix} \right\|_2 \leq \xi + \eta.$$

If we take $w = L^T x$, $\xi = 1$ and $\eta = -q^T x - r$, then inequality (2.16) is equivalent to the second order cone constraint

$$\|v\|_2 \leq v_0, \quad \text{where} \quad \begin{pmatrix} v_0 \\ v \end{pmatrix} = \begin{pmatrix} 1 - q^T x - r \\ 1 + q^T x + r \\ 2L^T x \end{pmatrix} \in \mathbb{R}^{h+2}. \tag{2.17}$$

Now the intersection of all such second order cone constraints can be easily expressed as Cartesian product of second order cones, thus we have obtained problem of SOCP in form 2.11.

2.3 Semidefinite programming

The semidefinite programming (SDP) is a convex optimization class which can be solved efficiently using interior point methods. In this section we will introduce the standard form of SDP and its dual. For reference and more information about this topic see [5] .

Firstly, let us introduce notation we will use to simplify the standard form.

Definition 2.18. Let A, X be real $n \times m$ matrices, we will denote their inner product

$$A \bullet X = Tr(A^T X).$$

Where $Tr(M)$ denotes trace of matrix M i.e. sum of the diagonal elements of M.

Definition 2.19 (SDP)**.** The primal–dual pair of the Semidefinite Program (SDP) in the standard form is

$$\begin{array}{llcll} & \textit{Primal} & & & \textit{Dual} \\ \text{minimize} & A_0 \bullet X, & & & \\ \text{subject to} & A_k \bullet X = b_k, & & \text{maximize} & b^T y, \\ & (k = 1, \dots, m), & & \text{subject to} & \sum_{k=1}^m y_k A_k + S = A_0, \\ & X \in \mathbb{S}^n_+, & & & S \in \mathbb{S}^n_+, \end{array} \tag{2.20}$$

where $X \in \mathbb{S}^n_+$, $y = (y_1, \dots, y_m)^T \in \mathbb{R}^m$ and $S \in \mathbb{S}^n$ are the variables; and symmetric matrices $A_0, A_1, \dots, A_m \in \mathbb{S}^n$ and scalars $b_1, \dots, b_m \in \mathbb{R}$ are given.

Surprisingly, the variable in SDP is a symmetric matrix (not a vector). In order to be consistent with other classes we will sometimes use a *svec* operator.

Definition 2.21. We define operator $svec : \mathbb{S}^n \to \mathbb{R}^{n(n+1)/2}$, such that for any $n \times n$ symmetric matrix M

$$svec(M) = (\delta_{11} M_{11}, \delta_{12} M_{12}, \delta_{22} M_{22}, \dots, \delta_{1n} M_{1n}, \dots, \delta_{nn} M_{nn})^T, \tag{2.22}$$

where

$$\delta_{ij} = \begin{cases} 1, & \text{if } \ i = j, \\ \sqrt{2}/2, & \text{otherwise.} \end{cases} \tag{2.23}$$

Notice that δ_{ij} are defined cleverly, so that inner product of symmetric matrices is equivalent to the standard inner product of their images

$$A \bullet X = svec(A)^T svec(X), \tag{2.24}$$

for any pair of symmetric matrices $A, X \in \mathbb{S}^n$. Now we can easily formulate the problems of SDP in terms of standard inner product over the space of real vectors.

$$\begin{array}{ll} \text{minimize} & svec(A_0)^T svec(X), \\ \text{subject to} & svec(A_k)^T svec(X) = b_k, \ (k = 1, \dots, m), \\ & svec(X) \in \mathcal{K}(\mathbb{S}^n_+), \end{array} \tag{2.25}$$

where $svec(X)$ is variable, $\mathcal{K}(\mathbb{S}^n_+) = \{svec(U) \in \mathbb{R}^{n(n+1)/2} \mid U \in \mathbb{S}^n_+\}$ and problem data are from the standard SDP (2.20).

2.3.1 Relation to previous classes

The SDP primal-dual pair looks suspiciously similar to the both LP and SOCP primal-dual pairs. The only difference between LP and SOCP is the nonnegative orthant is replaced by second order cone. The SDP, in the *svec*-operator form (2.25), further generalizes the cone constraint with the semidefinite cone.

In fact, SOCP is subclass of SDP. We will show how the standard SOCP can be rewritten as SDP. First of all, instead of minimizing $c^T x$ we will minimize t with additional constraint $t \geq c^T x$.

The only nontrivial part is to rewrite conic constraint

$$x \in Q_n \;\Leftrightarrow\; ||\bar{x}|| \leq x_1 \;\Leftrightarrow \left\{ \begin{array}{r} \bar{x}^T \bar{x} \leq x_1^2 \\ 0 \leq x_1 \end{array} \right\} \tag{2.26}$$

$$\Leftrightarrow \left\{ \begin{array}{r} \frac{\bar{x}^T \bar{x}}{x_1} \leq x_1 \\ 0 \leq x_1 \end{array} \right\} \Leftrightarrow \begin{pmatrix} x_1 & \bar{x}^T \\ \bar{x} & x_1 I_{n-1} \end{pmatrix} \succeq 0. \tag{2.27}$$

Where last equivalence is provided by Schur complement lemma (see appendix, Theorem A.1). From here, it can be easy brought to the standard form (see appendix).

In case of more general standard form of SOCP (2.11), the $x \in Q$ constraint can be transformed similarly.

$$x = (x^1, \ldots, x^k)^T \Leftrightarrow M = diag(M_1, \ldots, M_k) \succeq 0, \tag{2.28}$$

where M is a block diagonal matrix, with blocks M_i of the form (2.27), for $i = 1, \ldots, k$, corresponding to the constraints $x^i \in Q_{n_i}$.

2.4 Conic programming

The previous mentioned classes are quite similar. With respect to their variable space, all of them have linear objective, linear constraints and cone constraint.

In fact, they are special cases of the so called conic linear programs.

Definition 2.29 (Conic Programming)**.** The primal–dual pair of the Linear Conic Program in the standard form is

$$\begin{array}{llcll} \multicolumn{2}{c}{Primal} & & \multicolumn{2}{c}{Dual} \\ \text{minimize} & c^T x, & & \text{maximize} & b^T y, \\ \text{subject to} & Ax = b, & & \text{subject to} & A^T y + s = c, \\ & x \in \mathcal{K}, & & & s \in \mathcal{K}^*, \end{array} \tag{Conic Program}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$ and $s \in \mathbb{R}^n$ are the variables; and $m \times n$ real matrix A, vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, and the proper cone $\mathcal{K}$ are given problem data. The

$$K^* = \{z \mid \forall x \in \mathcal{K},\; x^T z \geq 0\}, \tag{2.30}$$

denotes the dual cone of $\mathcal{K}$ (see appendix A.2).

Conic programming contains, but is not limited to, any problems combined from LP, SOCP and SDP programs. For example

$$\begin{array}{ll} \text{minimize} & c^T x, \\ \text{subject to} & A^i x^i = b^i, \ (i = 1, \dots, k), \\ & x = (x^1, \dots, x^k) \in \mathcal{K} = (\mathcal{K}^1, \dots, \mathcal{K}^k), \end{array} \tag{2.31}$$

where the variable $x = (x^1, \dots, x^k)^T$ is the Cartesian product of the variables x^i, constrained by various LP, SOCP or SDP constraints $A^i x^i = b$, $x_i \in \mathcal{K}^i$, where each $\mathcal{K}^i$ is either nonegative orthant, second order cone or semidefinite cone.

This is due to the fact, that all cones we have talked about so far are proper cones (nonegative orthant $\mathbb{R}^n_+$, second order cone Q_n and semidefinite cone $\mathbb{S}^n_+$ as a subset of $\mathbb{R}^{n(n+1)/2}$). For proper cones it holds that their Cartesian product is again proper cone (see Section A.2 in appendix).

2.5 Dual problems

2.5.1 Dual problem of conic programming

We will derive the dual forms of LP, SOCP, SDP all at once by deriving Lagrange dual of general conic program

$$\begin{array}{ll} \text{minimize} & c^T x, \\ \text{subject to} & Ax = b, \\ & x \in \mathcal{K}. \end{array} \tag{2.32}$$

The Lagrangian of the problem is given by $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \times \mathcal{K}^* \to \mathbb{R}$,

$$\mathcal{L}(x, y, s) = c^T x + y^T (b - Ax) - s^T x. \tag{2.33}$$

The last term (notice that $s \in \mathcal{K}^*$) is added to take account of the conic constraint $x \in \mathcal{K}$. It is with negative sign in order to have $\mathcal{L}(x, \cdot, \cdot) \le c^T x$ for all x feasible in (2.32). Indeed, from the very definition of dual cone:

$$\sup_{s \in \mathcal{K}^*} -s^T x = \begin{cases} 0 & \text{if } x \in \mathcal{K}, \\ +\infty & \text{otherwise.} \end{cases} \tag{2.34}$$

Therefore, the Lagrange dual function is

$$\begin{aligned} g(y, s) &= \inf_x \ \mathcal{L}(x, y, s) && (2.35) \\ &= \inf_x \ y^T b + (c + A^T y - s)^T x && (2.36) \\ &= \begin{cases} b^T y & \text{if } c - A^T y - s = 0, \\ -\infty & \text{otherwise.} \end{cases} && (2.37) \end{aligned}$$

Hence, the dual problem of linear conic programming in the standard form is

$$\begin{array}{ll} \text{maximize} & b^T y \\ \text{subject to} & A^T y + s = c \\ & s \in \mathcal{K}^* \end{array} \tag{2.38}$$

Since $\mathbb{R}^n_+$, Q_n and $\mathbb{S}^n_+$ are self-dual, by replacing the $\mathcal{K}$ (and $\mathcal{K}^*$) with any of these cones, we get the dual of standard LP, SOCP and SDP as given in the Section 2.

Remark 2.39. For a closed cone $\mathcal{K}$ it holds that $\mathcal{K}^{**} = \mathcal{K}$ (where $\mathcal{K}^{**}$ denotes dual cone of the dual cone). Therefore, repeating the above procedure, one can easily show that dual of this dual problem is the original primal program (2.32).

2.5.2 Dual problem of QCQP

We will derive dual form of standard QCQP

$$\begin{array}{ll} \text{minimize} & x^T P_0 x + q_0^T x + r_0 \\ \text{subject to} & x^T P_k x + q_k^T x + r_k \leq 0, \ (k = 1, \dots, m) \end{array} \tag{2.40}$$

The Lagrangian of the problem is given by $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m_+ \to \mathbb{R}$,

$$\mathcal{L}(x, y) = x^T P_0 x + q_0^T x + r_0 + \sum_{k=1}^{m} y_k (x^T P_k x + q_k^T x + r_k) \tag{2.41}$$

$$= x^T P(y) x + q(y)^T x + r(y), \tag{2.42}$$

where

$$P(y) = P_0 + \sum_{k=1}^{m} y_k P_k, \quad q(y) = q_0 + \sum_{k=1}^{m} y_k q_k, \quad r(y) = r_0 + \sum_{k=1}^{m} y_k r_k. \tag{2.43}$$

It holds that $\inf_x \mathcal{L}(x, y) > -\infty$ if and only if $P(y) \succeq 0$ and there exists $\hat{x}$ such that $P(y)\hat{x} + q(y) = 0$.

Thus, the Lagrange dual function is

$$g(y) = \min_x \ \mathcal{L}(x, y) \tag{2.44}$$

$$= \begin{cases} -\frac{1}{4} q(y)^T P(y)^\dagger q(y) + r(y) & \text{if } P(y) \succeq 0, \quad q(y) \in \mathcal{R}(P(y)) \\ -\infty & \text{otherwise,} \end{cases} \tag{2.45}$$

where $P^\dagger$ denotes Moore-Penrose pseudoinverse of P (see appendix). Finally, dual form of standard QCQP problem is

$$\begin{array}{ll} \text{maximize} & -\frac{1}{4} q(y)^T P(y)^\dagger q(y) + r(y), \\ \text{subject to} & y \geq 0, \\ & P(y) \succeq 0, \\ & \mathcal{R}(q(y)) \subseteq \mathcal{R}(P(y)), \end{array} \tag{QCQP Dual}$$

where $y \in R^m$ is variable; and problem data $P_0, P_1 \dots, P_m, q_0, q_1 \dots, q_m, r_0, r_1 \dots, r_m$ are given from the primal QCQP above.

This dual problem is basically a SDP (in the LMI form). We first rewrite the objective as linear function t with additional constraint.

$$\begin{array}{ll} \text{maximize} & t, \\ \text{subject to} & t \leq -\frac{1}{4} q(y)^T P(y)^\dagger q(y) + r(y), \\ & y \geq 0, \\ & P(y) \succeq 0, \\ & \mathcal{R}(q(y)) \subseteq \mathcal{R}(P(y)). \end{array} \tag{2.46}$$

Due to the Schur complement lemma (see appendix, Theorem A.1) the above is equivalent to

$$\begin{array}{ll} \text{maximize} & t, \\ \text{subject to} & M := \begin{pmatrix} r(y) - t & \frac{1}{2}q(y)^T \\ \frac{1}{2}q(y) & P(y) \end{pmatrix} \succeq 0, \\ & y \geq 0, \end{array} \tag{2.47}$$

where the matrix M is easily expanded as

$$M = M_0 + \sum_{k=1}^{m} y_k M_k - tE \tag{2.48}$$

with

$$M_0 = \begin{pmatrix} r_0 & \frac{1}{2}q_0^T \\ \frac{1}{2}q_0 & P_0 \end{pmatrix}, \quad M_k = \begin{pmatrix} r_k & \frac{1}{2}q_k^T \\ \frac{1}{2}q_k & P_k \end{pmatrix}, \quad E = \begin{pmatrix} 1 & 0_n^T \\ 0_n & 0_{n\times n} \end{pmatrix}. \tag{2.49}$$

2.5.3 Second dual of QCQP

Now the whole process can be repeated. Create Lagrangian $\mathcal{L}_d : \mathbb{R} \times \mathbb{R}_+^m \times \mathbb{S}_+^{n+1} \times \mathbb{R}_+^m \to \mathbb{R}$,

$$\mathcal{L}_d(t, y, Y, u) = t + Y \bullet M_0 + \sum_{k=1}^{m} y_k Y \bullet M_k - tY \bullet E + u^T y, \tag{2.50}$$

where

$$Y = \begin{pmatrix} x_0 & x^T \\ x & X \end{pmatrix} \succeq 0. \tag{2.51}$$

The Lagrange dual function is

$$\begin{aligned} g_d(Y, u) &= \sup_{t,y} \mathcal{L}_d(t, y, Y, u) && (2.52) \\ &= \begin{cases} Y \bullet M_0, & \text{if } \left[\begin{array}{l} Y \bullet M_k \leq 0, \ k = 1, \ldots, m, \\ x_0 = 1 \ \text{ and } \ u = 0_m, \end{array} \right. \\ \infty, & \text{otherwise.} \end{cases} && (2.53) \end{aligned}$$

It is easy to see that with $x_0 = 1$ and Schur complement lemma we have

$$\begin{aligned} Y \bullet M_0 &= P_0 \bullet X + q_0^T x + r_0, && (2.54) \\ Y \bullet M_k \leq 0 &\Leftrightarrow P_k \bullet X + q_k^T x + r_k \leq 0, && (2.55) \\ Y \succeq 0 &\Leftrightarrow X \succeq xx^T. && (2.56) \end{aligned}$$

Thus we obtain the following SDP as second dual of QCQP

$$\begin{array}{ll} \text{minimize} & P_0 \bullet X + q_0^T x + r_0 \\ \text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m) \\ & X \succeq xx^T. \end{array} \tag{2.57}$$

Chapter 3

Relaxations

It was mentioned in the beginning that our strategy is to relax the nonconvex QCQPs (1.2) to easier problem. Let us first explore what is a relaxation and how can it be useful.

Relaxation is usually freely understood as an optimization problem which is obtained by relaxing (loosening) some constraints or even by approximating objective function with a different one. The goal is to obtain a problem which is easier to solve, but still carries some kind of information about the original one. For example, solving the relaxation may give an approximation of the original problem solution.

One could say, that relaxation of minimization problem

$$\begin{array}{ll} \text{minimize} & f(x), \\ \text{subject to} & x \in X, \end{array} \tag{3.1}$$

is another minimization problem

$$\begin{array}{ll} \text{minimize} & f_R(x), \\ \text{subject to} & x \in X_R, \end{array} \tag{3.2}$$

with properties $X \subseteq X_R$ and $c_R(x) \leq c(x) \forall x \in X$. It easily follows, that solving the relaxed problem will provide a lower bound on the optimal value of original problem. In some cases we can also extract a feasible solution of the original problem from solution of the relaxation. In that case we obtain an upper bound for the optimal value.

Moreover, these bounds may not only give us an idea about the optimal value, but also, may provide means to find an optimal solution of the original problem.

In the following we will explore the well known SDP relaxation which is basically casting the nonconvex QCQP into the convex SDP class. Furthermore, we will introduce some of the approaches for either loosening the SDP relaxation (in order to gain more speed) or strengthening these resulting SDP, SOCP, convex QP and LP relaxations (in order to obtain tighter bounds).

3.1 SDP relaxation of QCQP

Since Goemans and Williamson [15] proposed the SDP relaxation of the max-cut problem and proved its 0.878 approximation bound of the optimal value, a lot of work have been focused on solving the nonconvex (mostly combinatorial) QP problems using SDP relaxation methods. In this section we will derive the standard SDP relaxation of QCQP.

Consider the QCQP (1.2). Using identity $x^T P_k x = P_k \bullet xx^T$, which follows from the Definiton 2.18, it can be rewritten as follows

$$\begin{array}{ll} \text{minimize} & P_0 \bullet X + q_0^T x + r_0, \\ \text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m) \\ & X = xx^T. \end{array} \tag{3.3}$$

Notice, that the variable X is a symmetric $n \times n$ matrix. The problem can be reformulated in the following way:

$$\begin{array}{ll} \text{minimize} & M_0 \bullet Y, \\ \text{subject to} & M_k \bullet Y \leq 0, \ (k = 1, \ldots, m), \\ & X = xx^T, \end{array} \tag{3.4}$$

where ,

$$M_k = \begin{pmatrix} \alpha_k & \frac{1}{2} q_k^T \\ \frac{1}{2} q_k & P_k \end{pmatrix}, \qquad Y = \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix}. \tag{3.5}$$

This problem still has a non-convex constraint $X = xx^T$, which can be relaxed by a convex constraint, as stated in the following lemma.

Lemma 3.6. *Let $x \in \mathbb{R}^n$, an $n \times n$ symmetric matrix X, and $n+1 \times n+1$ symmetric matrix Y such that*

$$Y = \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix}.$$

Then

(i) $X \succeq xx^T$ if and only if $Y \succeq 0$.

(ii) $X = xx^T$ holds if and only if $Y \succeq 0$ and $\operatorname{rank} Y = 1$.

Proof. (i) The statement follows from Schur complement lemma for PSD (see Theorem A.1 in the appendix)

(ii) ($\Rightarrow$) If $X = xx^T$, then also $X \succeq xx^T$, thus $Y \succeq 0$ holds by (i). And since $X = xx^T$,

$$Y = \begin{pmatrix} 1 & x^T \\ x & xx^T \end{pmatrix} = \begin{pmatrix} 1 \\ x \end{pmatrix} (1, \ x^T).$$

Hence $\operatorname{rank} Y = 1$.
($\Leftarrow$) Let $Y \succeq 0$ and $\operatorname{rank} Y = 1$. Since $\operatorname{rank} Y = 1$, each row of Y must be scalar multiple of the first (obviously nonzero) row $(1, \ x^T)$. To match the first column the $(i+1)$-st row of Y must be $x_i(1, \ x^T)$, for $i = 1, \ldots, n$. Therefore, $X = xx^T$. □

Remark 3.7. Notice that we have proven last implication of (ii) without using $Y \succeq 0$. In fact it is redundant. It also holds that $X = xx^T \Leftrightarrow \operatorname{rank} Y = 1$. In fact, there are only 2 options for Y of rank 1: $Y = vv^T$ or $Y = -vv^T$. The second option is easily excluded, because $Y_{11} = 1 > 0$. However, this redundant constraint $Y \succeq 0 \Leftrightarrow X \succeq xx^T$ will let us keep something from the rank 1 constraint after relaxing it. This approach of adding the redundant constraints (also known as valid inequalities) is often usefull for strengthening the relaxation. For more on valid inequalities see [12, 13, 14].

Using the Lemma 3.6 we obtain another equivalent formulation of the original QCQP

$$\begin{array}{ll} \text{minimize} & M_0 \bullet Y, \\ \text{subject to} & M_k \bullet Y \leq 0, \;\; (k = 1, \dots, m) \\ & Y \succeq 0, \quad \operatorname{rank} Y = 1. \end{array} \tag{3.8}$$

relaxing the nonconvex rank 1 constrain we get the following SDP relaxation of (1.2)

$$\begin{array}{ll} \text{minimize} & M_0 \bullet Y, \\ \text{subject to} & M_k \bullet Y \leq 0, \;\; (k = 1, \dots, m) \\ & Y \succeq 0. \end{array} \tag{3.9}$$

Expanding the terms $M_k \bullet Y$ we obtain the standard SDP relaxation

$$\begin{array}{ll} \text{minimize} & P_0 \bullet X + q_0^T x + r_0, \\ \text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \;\; (k = 1, \dots, m) \\ & X \succeq xx^T. \end{array} \tag{3.10}$$

Remark 3.11. Notice that this is exactly the second dual of QCQP (2.57).

The above relaxed problem has different variable space $(X, x) \in \mathbb{R}^{n \times n} \times \mathbb{R}^n$ than original problem $x \in \mathbb{R}^n$. In other words, the variable space increased from $O(n)$ to $O(n^2)$ variables.

3.2 Convex QP and SOCP relaxation

These relaxations are expected to provide weaker bounds in less time compared to SDP relaxations. In fact, SOCP relaxations are often constructed as further relaxations of SDP relaxations. So, in certain sense, one can see that SOCP relaxations are never tighter than their SDP counterparts. [1]

In this section we will introduce some approaches for further loosening of SDP relaxation. There are 2 main reasons why SDP is expensive to solve. The $O(n^2)$ variable space and semidefinite constraint.

Using the procedure from section 2.2, any convex instance of QCQP can be formulated as SOCP. Therefore we can consider any convex quadratic relaxation as SOCP relaxation. Specifically, a convex QCQP relaxation may be represented as

$$\begin{array}{ll} \text{minimize} & x^T B_0 x + b_0^T x + \beta_0 \\ \text{subject to} & x^T B_k x + b_k^T x + \beta_i \leq 0, \;\; (k = 1, \dots, l) \end{array} \tag{3.12}$$

where all $B_k \succeq 0$ for $k = 0, \ldots, l$. We say that (3.12) is SOCP relaxation of QCQP (1.2) if any x feasible in (1.2) is also feasible in (3.12) and $x^T B_0 x + b_0^T x + \beta_0 \leq x^T P_0 x + q_0^T x + r_0$ holds.

In fact, one possible approach for forming such a SOCP (or convex QP) relaxation is further relaxing the SDP relaxation

$$\begin{array}{ll} \text{minimize} & P_0 \bullet X + q_0^T x + r_0, \\ \text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m) \\ & X \succeq xx^T. \end{array} \tag{3.13}$$

3.2.1 SOCP relaxation of semidefinite constraint

Semidefinite constraint $X - xx^T \succeq 0$ in (3.13) is equivalent to $C \bullet (X - xx^T) \geq 0$ for all $C \in S_+^n$. Using this fact, authors of [4] propose SOCP relaxation of the semidefinite constraint $X - xx^T \succeq 0$ by replacing it with multiple constraints of the form

$$x^T C_i x - C_i \bullet X \leq 0 \quad (i = 1, \ldots, t).$$

Since $C_i \succeq 0$, these are convex quadratic constraints and using the procedure described earlier (in the Section 2.2.1) one can formulate them equivalently as a second order cone constraints of the form

$$\begin{pmatrix} v_0^i \\ v^i \end{pmatrix} = \begin{pmatrix} 1 + C_i \bullet X \\ 1 - C_i \bullet X \\ 2L_i^T x \end{pmatrix}, \quad ||v^i|| \leq v_0^i, \ \ (i = 1, \ldots, l), \tag{3.14}$$

where L_i is obtained from the Choelsky decomposition of $C_i = L_i L_i^T$.

They also show how to extract these convex inequalities from original quadratic inequality constraints. We will omit indices and consider the constraint $x^T P x + q^T x + r \leq 0$. Let

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_l \geq 0 > \lambda_{l+1} \geq \cdots \geq \lambda_n, \tag{3.15}$$

be the eigenvalues of the matrix P and let $u_1, \ldots, u_n$ be the corresponding eigenvectors, such that

$$\|u_i\|_2 = 1 \ (i = 1, \ldots, n) \quad \text{and} \quad u_i^T u_j = 0 \ (i \neq j). \tag{3.16}$$

Then $P = \sum_{j=1}^h \lambda_j u_j u_j^T$, and any matrix of the either form

$$C^+ \ = \ \sum_{j \in J} \lambda_j u_j u_j^T \quad \text{for } J \subseteq \{1, \ldots, l\}, \tag{3.17}$$

$$C^- \ = - \sum_{j \in J} \lambda_j u_j u_j^T \quad \text{for } J \subseteq \{l+1, \ldots, n\}, \tag{3.18}$$

is positive semidefinite and can be used in $x^T C x - C \bullet X \leq 0$ constraint.

The cost of solving the resulting SOCP depends very much on the ranks of $C_i \in \mathbb{S}_+^n$, the larger their ranks are, the more auxiliary variables we need to introduce and the more expensive the cost of solving the resulting SOCP becomes. In an attempt to keep the amount of computation small, low rank C_i are reasonable. Authors of [4] suggest putting $C_i = e_i e_i^T$, which corresponds to the constraints

$$x_i^2 - X_{ii} \leq 0. \tag{3.19}$$

They also employ rank-1 convex quadratic inequalities tied with the problem data, taking $C_i = u_i u_i^T$. where u_i are chosen as eigenvectors of the matrices P_k from the quadratic inequality constraints of original QCQP. This choice corresponds to constraints

$$x^T u_i u_i^T x - u_i u_i^T \bullet X \leq 0 \ (i = 1, \ldots, l), \tag{3.20}$$

3.2.2 SOCP in the original variable

The SOCP relaxation introduced above has avoided the semidefinite constraint but still includes a matrix variable X from the SDP which increases size of the problem dramatically (from n to $n + n(n+1)/2$). In [4] authors also introduce a general relaxation scheme to obtain SOCP relaxation of QCQP (1.2) in the original variable space $x \in \mathbb{R}^n$. First they assume that objective function is linear (otherwise we can add new variable $t \geq x^T P_0 x + q_0^T + r_0$ and then minimize t). Then each P_k is written as

$$P_k = P_k^+ - P_k^-, \quad \text{where} \quad P_k^+, P_k^- \succeq 0, \quad k = 1, \ldots, m.$$

So that each constraint can be expressed as

$$x^T P_k^+ x + q_k^T x + r_k \leq x^T P_k^- x. \tag{3.21}$$

Then an auxiliary variable $z_k \in \mathbb{R}$ is introduced to represent $x^T P_k^- x = z_k$, but also immediately relaxed as $x^T P_k^- x \leq z_k$, resulting in covnex system

$$\begin{aligned} x^T P_k^+ x + q_k^T x + r_k &\leq z_k \\ x^T P_k^- x &\leq z_k. \end{aligned} \tag{3.22}$$

Finally, z_k must be bounded in some fashion, say as $z_k \leq \mu \in \mathbb{R}$, or else the relaxation would be useless. In the next section we will show an example of such bounds for z_k. In this way convex QCQP relaxation is constructed and it is simply transformed to SOCP using the procedure from [4] described in the Section 2.2.

3.2.3 Relaxing the semidefinite constraint and reducing the number of variables

The idea of combining both above approaches (introduced in [4]) is to relax each quadratic inequality constraint together with constraints of the form $x^T C x - C \bullet X \leq 0$ relaxing the semidefinite constraint. We will omit the indices and consider the constraint

$$x^T P x + q^T x + r \leq 0. \tag{3.23}$$

Let the eigenvalues λ_i and eigenvectors u_i of P for $(i = 1, \dots, n)$ be as above, see (3.15) and (3.16). And let $P = P^+ - P^-$ with

$$P^+ = \sum_{j=1}^{l} \lambda_j u_j u_j^T \succeq 0 \quad , \quad P^- = -\sum_{j=l+1}^{n} \lambda_j u_j u_j^T \succeq 0, \tag{3.24}$$

then we may rewrite the quadratic inequality constraint $x^T P x + q^T x + r \leq 0$ as

$$\begin{array}{l} x^T P^+ x + \sum_{j=l+1}^{n} \lambda_j z_j + q^T x + r \leq 0, \\ x^T (u_j u_j^T) x - z_j = 0, \ (j = l+1, \dots, n., \end{array} \tag{3.25}$$

Now, relaxing the last $n - l$ inequalities we obtain a set of convex inequalities

$$\begin{array}{l} x^T P^+ x + \sum_{j=p+1}^{h} \lambda_j z_j + q^T x + r \leq 0 \\ x^T (u_j u_j^T) x - z_j \leq 0, \ (j = p+1, \dots, h) \end{array} \tag{3.26}$$

It is necessary to add the appropriate constraints on the variables z_j to bound them from above, authors show that $\sum_{j=l+1}^{n} z_j \leq \|x\|_2^2$ follows from (3.25). In fact, if x and z_j for $(j = l+1, \dots, n)$ satisfy (3.25) then

$$\sum_{j=l+1}^{n} z_j = \sum_{j=l+1}^{n} x^T (u_j u_j^T) x \leq x^T \left(\sum_{j=1}^{n} u_j u_j^T \right) x \leq \|x\|_2^2, \tag{3.27}$$

where the last inequality holds because $\left(\sum_{j=1}^{n} u_j u_j^T \right)$ is an orthonormal matrix.

So the final relaxation of the $x^T P x + q^T x + r \leq 0$ constraint is

$$\begin{array}{l} x^T P^+ x + \sum_{j=p+1}^{h} \lambda_j z_j + q^T x + r \leq 0 \\ x^T (u_j u_j^T) x - z_j \leq 0, \ (j = p+1, \dots, h) \\ \sum_{j=l+1}^{n} z_j \leq \|x\|_2^2 \end{array} \tag{3.28}$$

In the end authors of [4] compare this relaxation and the one obtained by relaxing the semidefinite constraint without reducing the number of variables. Consider the following relaxation of the same quadratic constraint (3.23) together with valid inequalities relaxing the semidefinite constraint

$$\begin{array}{l} P \bullet X + q^T x + r \leq 0, \\ x^T P^+ x - P^+ \bullet X \leq 0, \\ x^T u_j u_j^T x - u_j u_j^T \bullet X \leq 0, \ (j = p+1, \dots, h). \end{array} \tag{3.29}$$

Suppose $(x, X) \in \mathbb{R}^n \times \mathbb{S}^n$ satisfy this relaxation, then x and $z_j = u_j^T X u_j$ for $j = p+1, \dots, h$ satisfy (3.28). So we may see (3.28) as further relaxation of (3.29). Therefore it will provide weaker bounds, however it will take less time to solve because of the smaller variable space. For further details about this approach see [3, 4].

3.3 Mixed SOCP-SDP relaxation

In [1] authors have introduced compromise, relaxation of the QCQP (1.2) somewhere between SDP and SOCP. We will describe their approach with the special case they provide as introduction. We are dealing with (1.2)

$$\begin{array}{ll} \text{minimize} & x^T P_0 x + q_0^T x + r_0 \\ \text{subject to} & x^T P_k x + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m) \end{array} \tag{3.30}$$

Let $\lambda_{min}(P_k)$ denote smallest eigenvalue of P_k. For all $k = 0, \ldots, m$ define $\lambda_k = -\lambda_{min}(P_k)$ so that $P_k + I\lambda_k \succeq 0$. Then (1.2) is equivalent to

$$\begin{array}{ll} \text{minimize} & -\lambda_0 x^T x + x^T(P_0 + \lambda_0 I)x + q_0^T x + r_0 \\ \text{subject to} & -\lambda_k x^T x + x^T(P_k + \lambda_k I)x + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m) \end{array} \tag{3.31}$$

which has following SOCP-SDP relaxation

$$\begin{array}{ll} \text{minimize} & -\lambda_0 Tr(X) + x^T(P_0 + \lambda_0 I)x + q_0^T x + r_0 \\ \text{subject to} & -\lambda_k Tr(X) + x^T(P_k + \lambda_k I)x + q_k^T x + r_k \leq 0, \\ & (k = 1, \ldots, m) \\ & X \succeq xx^T \end{array} \tag{3.32}$$

Notice that other than $X \succeq xx^T$, the only variables in X to appear in the program are diagonal elements X_{jj}. Also when $\lambda_k > 0$, one can see that with fixed x the diagonal entries of X can be made arbitrarily large to satisfy all constraints. As well when $\lambda_0 > 0$, an arbitrary large diagonal entry of X will drive the objective to $-\infty$. Therefore, in general, X_{jj} should be bounded to form a sensible relaxation. In the paper [1] they suppose that $x_j \in [0, 1]$ and use $X_{jj} \leq x_j$ to establish boundedness.

Remark 3.33. In general, if the feasible region is bounded, then there can be introduced box constraints $l_j \leq x_j \leq u_j$ and X_{jj} can be bounded for example by multiplying these two inequalities

$$\left.\begin{array}{r} x_j - l_j \geq 0 \\ u_j - x_j \geq 0 \end{array}\right\} \ x_j u_j + x_j l_j - l_j u_j \geq X_{jj}, \tag{3.34}$$

where $x_j x_j$ was replaced by X_{jj}. This is valid since equality $X = xx^T$ holds for every solution of the original problem.

Next proposition from [11] gives equivalent formulation of $X \succeq xx^T$ constraint, only in terms of x and diagonal entries of X.

Proposition 3.35. *Given a vector x and scalars $X_{11}, \ldots, X_{nn}$, there exists a symmetric-matrix completion $X \in S^n$ of $X_{11}, \ldots, X_{nn}$ satisfying $X \succeq xx^T$ if and only if $X_{jj} \geq x_j^2$ for all $j = 1, \ldots, n$.* [11]

Thus, in light of this proposition, the problem with additional bounding constraints $X_{jj} \leq x_j$, the problem (3.32) is equivalent to

$$\begin{array}{ll} \text{minimize} & -\lambda_0 Tr(X) + x^T(P_0 + \lambda_0 I)x + q_0^T x + r_0 \\ \text{subject to} & -\lambda_k Tr(X) + x^T(P_k + \lambda_k I)x + q_k^T x + r_k \leq 0, \\ & (k = 1, \ldots, m) \\ & x_j^2 \leq X_{jj} \leq x_j \quad (j = 1, \ldots, n) \end{array} \tag{3.36}$$

Compared to SDP relaxation (3.10), which has $O(n^2)$ variables, problem (3.36) has only $O(n)$ and hence is much faster to solve. On the other hand bound should be generally weaker than the SDP bound.

This approach is further generalized and explored in [1].

3.4 LP relaxation of QCQP

Semidefinite program relaxations can provide tight bounds, but they can also be expensive to solve by classical interior point methods (because of the $O(n^2)$ variables and semidefinite constraint).

Many researchers have studied different types of relaxations, for example, ones based on LP or SOCP. As we have seen in the first chapter, SDP is the broadest of the introduced classes, therefore the SDP relaxation will be generally stronger than LP or SOCP relaxation. However the advantage of LP may be the speed.

Simple LP relaxation may be obtained from SDP relaxation (3.10) replacing the semidefinite constraint $X \succeq xx^T$ with symmetry constraint $X = X^T$.

$$\begin{array}{ll} \text{minimize} & P_0 \bullet X + q_0^T x + r_0 \\ \text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \;\; (k = 1, \ldots, m) \\ & X = X^T. \end{array} \tag{3.37}$$

Suppose that the feasible set in (3.37) is bounded and we can formulate box constraints for each variable - either explicitly included (i.e. $P_k = 0$ for some k) or implied from the quadratic constraints (for example in combinatorial problems where constraint $x_i \in \{0, 1\}^n$ is included as $x_i(x_i - 1) = 0$ we may also include $0 \leq x_i \leq 1$ box constraint). We will separate all such box constraints in the next formulation.

$$\begin{array}{ll} \text{minimize} & P_0 \bullet X + q_0^T x + r_0 \\ \text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \;\; (k = 1, \ldots, m) \\ & l_i \leq x_i \leq u_i, \;\; (i = 1, \ldots, n) \\ & X = X^T. \end{array} \tag{3.38}$$

This is also referred to as lift and project LP relaxation.

3.4.1 Reformulation Linearization Technique (RLT)

The optimal value of (3.38) is usually weak lower bound as no constraint links the values of x and X variables. The main approach to provide these links and strengthen the relaxation is the Reformulation Linearization Technique (RLT) relaxation [18,16,17]. It adds linear inequalities to (3.38). These inequalities are derived from the variable bounds and constraints of the original problem as follows: multiply together two original constraints or bounds and relax each product term $x_i x_j$ with the variable X_{ij}. Note that this will be a valid inequality, since the original constraint $X = xx^T$ implies $x_i x_j = X_{ij}$. For instance, let x_i, x_j , $i, j \in \{1, 2, \ldots, n\}$

be two variables from (3.38). By taking into account only the four original bounds $x_i - l_i \geq 0$, $x_i - u_i \leq 0$, $x_j - l_j \geq 0$, $x_j - u_j \leq 0$, we get the RLT inequalities

$$\begin{array}{lcl} X_{ij} - u_i x_j - u_j x_i & \geq & -u_i u_j, \\ X_{ij} - u_i x_j - l_j x_i & \leq & -u_i l_j, \\ X_{ij} - l_i x_j - u_j x_i & \leq & -l_i u_j, \\ X_{ij} - l_i x_j - l_j x_i & \geq & -l_i l_j. \end{array} \tag{3.39}$$

Note that these constraints also hold when $i = j$, in which case the upper bounds are identical. Denote $l = (l_1, \ldots, l_n)^T$ and $u = (u_1, \ldots u_n)^T$. Using the vector and matrix inequalities (meaning that inequality holds in each coordinate) the resulting RLT relaxation can be written as

$$\begin{array}{ll} \text{minimize} & P_0 \bullet X + q_0^T x + r_0 \\ \text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \;\; (k = 1, \ldots, m) \\ & X - l x^T - x l^T \;\; \geq \;\; -l l^T, \\ & X - u x^T - x u^T \;\; \geq \;\; -u u^T, \\ & X - l x^T - x u^T \;\; \leq \;\; -l u^T, \\ & l \leq x \leq u, \quad X = X^T, \end{array} \tag{3.40}$$

The both upper bounds for X_{ij} are here together expressed in the single vector constraint due to symmetry. In fact, it is known that the original bound constraints $l \leq x \leq u$ are redundant and could be removed. If the QCQP contains linear constraints other than simple box constraints (i.e. $P_k = 0$ for some k) then additional constraints on X can be imposed.

3.4.2 Positive semidefinite cuts

In the above LP relaxations we have relaxed nonconvex $X = xx^T$ constraint in (3.3) with simple symmetry and later added RLT constraints to link the variables x and X. In fact we can go further in strengthening the LP relaxation.

Let us begin with SDP relaxation in the form (3.9), i.e.

$$\begin{array}{ll} \text{minimize} & M_0 \bullet Y \\ \text{subject to} & M_k \bullet Y \leq 0, \;\; (k = 1, \ldots, m) \\ & Y \succeq 0, \end{array} \tag{3.41}$$

where

$$M_k = \begin{pmatrix} \alpha_k & \frac{1}{2} q_k^T \\ \frac{1}{2} q_k & P_k \end{pmatrix}, \qquad Y = \begin{pmatrix} 1 & x^T \\ x & X \end{pmatrix}. \tag{3.42}$$

From the definition of positive semidefinite matrix it holds that

$$Y \succeq 0 \;\Leftrightarrow\; v^T Y v \geq 0, \;\forall v \in \mathbb{R}^{n+1}. \tag{3.43}$$

Notice that inequalities $v^T Y v \geq 0$ are linear in Y therefore linear in both x and X, we will refer to the inequalities in this form as PSD cuts. In order to obtain a linear program, the semidefinite constraint cannot be replaced by the infinite number of constraints $v^T Y v \geq 0$. However, it is possible to relax it with the finite number of these linear inequalities.

$$\begin{array}{ll} \text{minimize} & M_0 \bullet Y \\ \text{subject to} & M_k \bullet Y \leq 0, \ (k = 1, \ldots, m) \\ & v^T Y v \geq 0, \ v \in V, \end{array} \tag{3.44}$$

for some finite set of vectors V.

The usual approach is declaring such PSD cuts with regard to a feasible solution $\bar{Y}$ of the relaxation (3.44) (for example $\bar{Y}$ may be the optimal solution of (3.44)). If $\bar{Y}$ is not positive semidefinite, then such vectors v can be added to V, that will cut off $\bar{Y}$ from the feasible set by adding the inequalities $v^T Y v \geq 0$ - that is why it is called PSD cut.

Hopefully, we would like to restrict the feasible set this way to tighten the gap between the optimal value of the LP relaxation (3.44) and the optimal value of the SDP relaxation. For example by cutting off the optimal solutions of the actual LP relaxation.

The vectors v may be chosen as eigenvectors corresponding to negative eigenvalues of an arbitrary matrix $\bar{Y}$. In [17] authors note two weaknesses of such approach. Firstly, only one cut is obtained from each eigenvector, while computing the spectral decomposition requires non trivial investment of time. Secondly, such cuts are usually very dense, i.e. almost all entries of $\bar{v}\bar{v}^T$ are nonzero, adding such inequalities might considerably slow down the computation.

They propose an efficient algorithm to generate sparse cuts from given matrix $\bar{Y}$ and initial vector $\bar{v}$ with $\bar{v}^T \bar{Y} \bar{v} \leq 0$,.

TODO: it might be interesting to find out what is the good choice of such $\bar{Y}$ and $\bar{v}$. (1 - based on problem data, i.e. matrices from quadratic inequalities, 2 - iteratively, always take $\bar{Y}$ as optimal solution of previous relaxation and strengthen by cutting it off, until we reach precision of SDP bound)

3.5 Strengthening the SDP relaxation

3.5.1 SDP + RLT relaxation

It is not surprising that adding RLT valid inequalities into LP relaxation will significantly strengthen the relaxation. On the other hand, it is quite surprising that RLT inequalities may also help when added to SDP relaxation. In fact Anstreicher in [16] showed that following RLT + SDP relaxation is stronger than both SDP or

RLT relaxations.

$$
\begin{array}{ll}
\text{minimize} & P_0 \bullet X + q_0^T x + r_0 \\
\text{subject to} & P_k \bullet X + q_k^T x + r_k \leq 0, \ (k = 1, \ldots, m) \\
& X - lx^T - xl^T \quad \geq \quad -ll^T, \\
& X - ux^T - xu^T \quad \geq \quad -uu^T, \\
& X - lx^T - xu^T \quad \leq \quad -lu^T, \\
& l \leq x \leq u, \quad X \succeq xx^T.
\end{array}
\tag{3.45}
$$

However, in terms of computational cost, in his experiments each RLT or SDP bound for the tested problems required approximately 1 second of computation, but each SDP+RLT bound required over 200 seconds of computation. It is well known that "mixed" SDP/LP problems involving large numbers of inequality constraints are computationally challenging, and reducing the work to solve such problems is an area of ongoing algorithmic research. In [19] they propose enhancing the RLT relaxation with PSD cuts as linear relaxations of the semidefinite constraint. Also adding a SOCP relaxation of semidefinite constraint into lift and project LP relaxation proposed in [4] offers a way to avoid the cost of combining SDP and LP constraints.

3.5.2 Best D.C. decompositions

TODO: reconsider completing or deleting this subsection

Another interesting approach for strengthening the SDP relaxation was proposed in [20]. They restrict to the problems with convex quadratic constraints and nonconvex objective, i.e. the matrices $P_1, \ldots, P_k$ are assumed to be positive semidefinite.

$$P_k \succeq 0 \ \text{ for } \ k = 1 \ldots m. \tag{3.46}$$

Let v_i for $i = 1, \ldots, p$ be nonzero vectors in $\mathbb{R}^n$ and suppose that there exists $\lambda \in \mathbb{R}^p_+$ and $\mu \in \mathbb{R}^m_+$ such that

$$P_0 + \sum_{i=1}^{p} \lambda_i v_i v_i^T - \sum_{i=1}^{m} \mu_i P_i \succeq 0, \tag{3.47}$$

Then they decompose the nonconvex objective $f_0(x) = x^T P_0 x + q_0^T x + r_0$ as D.C. (difference of convex functions)

$$x^T \left(P_0 + \sum_{i=1}^{p} \lambda_i v_i v_i^T + \sum_{i=1}^{m} \mu_i P_i \right) x - \sum_{i=1}^{p} \lambda_i (v_i^T x)^2 - \sum_{i=1}^{m} \mu_i x^T P_i x + q_0^T x + r_0. \tag{3.48}$$

After that they convexify the objective by underestimating the nonconvex terms $-\sum_{i=1}^{p} \lambda_i (v_i^T x)^2$ and $-\sum_{i=1}^{m} \mu_i x^T P_i x$ with a (convex) piecewise linear function,

and show that optimal choice of parameters λ, μ can be reduced to semidefinite program and show that the resulting SDP has dual equivalent to standard SDP (3.10) strengthened with additional valid inequalities.

Chapter 4

Combinatorial optimization problems

In this chapter we will first talk about max-cut problem, since its tight bound for semidefinite relaxation was the initial motivation for the research of semidefinite relaxations, algorithms for semidefinite programming and convex relaxations of other combinatorial problems.

Next we will introduce the branch and bound framework, which provides a way to solve (not only) combinatorial problems to optimality using bounds obtained by solving relaxed problems.

4.1 The maximum cut problem

In this chapter we are going to explore one of the classical problems of combinatorial optimization - the maximum cut problem. First we will state the problem and show that it can be formulated as QCQP. Then we will describe in detail the famous analysis by Goemans and Williamson [15] showing its SDP relaxation gives 0.878 approximation bound.

Problem statement (max-cut). Let $G = (V, \mathcal{E})$ be an undirected graph where $V = \{1, \dots, n\}$ and $\mathcal{E}$ are the sets of vertices and edges, respectively. We assume that a weight w_{ij} is attached to each edge $[i, j] \in \mathcal{E}$. For a partition $(S, \bar{S})$ of V, i.e.

$$(S, \bar{S}), \quad \text{s.t.} \quad S \cup \bar{S} = V \quad \wedge \quad S \cap \bar{S} = \emptyset, \tag{4.1}$$

we define

$$w(S, \bar{S}) = \sum_{[i,j] \in \mathcal{E}, i \in S, j \in \bar{S}} w_{ij}. \tag{4.2}$$

The maximum cut problem (or shortly max-cut) is to find a partition maximizing $w(S, \bar{S})$.

4.1.1 QCQP formulation of the max-cut

Let us assign zero weight to all the non edges $w_{ij} = 0$ for $[i, j] \notin \mathcal{E}$. For each $i \in V$, we put

$$x_i = \begin{cases} 1 & \text{if } i \in S, \\ -1 & \text{if } i \in \bar{S}. \end{cases} \tag{4.3}$$

Because $\frac{1}{2}(1 - x_i x_j) = 1$ if i and j belong to different partitions and 0 otherwise, we see that problem is equivalent to

$$\begin{array}{ll} \text{maximize} & \frac{1}{2}\sum_{i<j} w_{ij}(1 - x_i x_j), \\ \text{subject to} & x \in \{-1, 1\}^n. \end{array} \tag{4.4}$$

This can be easily rewritten in matrix form. Let W be a $n \times n$ symmetric matrix of weights with entries $W_{ij} = w_{ij}$, and let

$$L = diag(We) - W, \tag{4.5}$$

where e is all ones vector and $diag(We)$ is the diagonal matrix with diagonal We. Then, using $x_i^2 = 1$ we have

$$\begin{aligned} w(S, \bar{S}) &= \frac{1}{2}\sum_{i<j} w_{ij}(1 - x_i x_j) && (4.6) \\ &= \frac{1}{4}\sum_{[i,j] \in V^2} w_{ij}(1 - x_i x_j) && (4.7) \\ &= \frac{1}{4}x^T(diag(We) - W)x && (4.8) \\ &= \frac{1}{4}x^T L x. && (4.9) \end{aligned}$$

Now, with $x_i \in \{0, 1\} \Leftrightarrow x_i^2 = x^T e_i e_i^T x = 1$ constraint, we obtain a following problem

$$\begin{array}{ll} \text{maximize} & \frac{1}{4}x^T L x, \\ \text{subject to} & x^T e_i e_i^T x = 1, \ (i = 1, \ldots, n). \end{array} \tag{4.10}$$

Therefore max-cut is indeed an instance of QCQP.

4.1.2 The Goemans and Williamson analysis

In their famous article [15] authors first relax the problem (4.4) by allowing x_i to be represented by vector variables on the unit sphere $v_i \in S^n$ for $i \in V$. And define objective function by replacing the $x_i x_j$ with scalar products $v_i^T v_j$. In this way, the objective reduces to $\frac{1}{2}\sum_{i<j} w_{ij}(1 - x_i x_j)$ when all the v_i are lying in 1-dimensional space. The resulting relaxation is

$$\begin{array}{ll} \text{maximize} & \frac{1}{2}\sum_{i<j} w_{ij}(1 - v_i^T v_j), \\ \text{subject to} & v_i \in S^n \ (i = 1, \ldots, n). \end{array} \tag{4.11}$$

We will show later that this relaxation can be solved using the semidefinite programming.

They propose simple randomized algorithm, which we will refer to as GW-algorithm

1. Solve (4.11), obtaining an optimal set of vectors v_i.
2. Generate r uniformly distributed on the unit sphere S^n.
3. Set $S = \{i \mid v_i^T r \geq 0\}$.

In other words in step 2, the random hyperplane through the origin is chosen. In step 3 we form a partitions based on the separation of the vectors v_i by the hyperplane. The sets S and $\bar{S}$ are formed by indices of vectors lying "above" and "bellow" the hyperplane, respectively.

Proposition 4.12. *Let ω be the value of the cut produced by the above algorithm and $E[\omega]$ its expected value. Authors show that*

$$E[\omega] = \frac{1}{\pi} \sum_{i<j} w_{ij} \arccos(v_i^T v_j), \tag{4.13}$$

and

$$E[\omega] \geq \alpha \frac{1}{2} \sum_{i<j} w_{ij} (1 - v_i^T v_j) \quad \textit{for} \quad \alpha = 0.87856. \tag{4.14}$$

Proof. First we will show (4.13). By definition and linearity of expected value in the first step, using symmetry in the second we have

$$\begin{aligned} E[\omega] &= \sum_{i<j} w_{ij} \; Pr\left[sgn(v_i^T r) \neq sgn(v_j^T r)\right] & (4.15) \\ &= \sum_{i<j} w_{ij} \; 2Pr\left[v_i^T r \geq 0 \;\wedge\; v_j^T r < 0\right]. & (4.16) \end{aligned}$$

Fix the i, j and let $\theta = \arccos(v_i^T v_j)$ be the angle between v_i and v_j. In the last expression, there is a probability that v_i is above the hyperplane with normal vector r and v_j is bellow. The set $\{r \mid v_i^T r \geq 0 \;\wedge\; v_j^T r < 0\}$ corresponds to the intersection of two half-spaces whose dihedral angle is precisely θ its intersection with the sphere is a spherical digon of angle θ and, by symmetry of the sphere, thus has measure equal to $\theta/2\pi$ times the measure of the full sphere. In other words,

$$Pr\left[v_i^T r \geq 0 \;\wedge\; v_j^T r < 0\right] = \frac{\theta}{2\pi} = \frac{\arccos(v_i^T v_j)}{2\pi}. \tag{4.17}$$

Plugging this identity into (4.16), the first part follows.

Secondly we will show (4.14). It is enough to prove that

$$\frac{1}{\pi} \arccos(v_i^T v_j) \geq \alpha \frac{1}{2} (1 - v_i^T v_j), \tag{4.18}$$

because multiplying by w_{ij} and summing up these inequalities yelds

$$E[\omega] = \frac{1}{\pi}\sum_{i<j} w_{ij} \arccos(v_i^T v_j) \geq \alpha\frac{1}{2}\sum_{i<j} w_{ij}(1 - v_i^T v_j), \tag{4.19}$$

where the left hand side is equal to $E[\omega]$ from (4.13).

Since $v_i, v_j \in S^n$ are unit vectors, the inner product is bounded by $v_i^T v_j \in [-1, 1]$. Thus, there exists $\theta \in [0, \pi]$ such that $v_i^T v_j = \cos(\theta)$. Rearranging terms and substituting $v_i^T v_j = \cos(\theta)$ we can see that (4.18) reduces to

$$0.87856 = \alpha \leq \min_{\theta\in[0,\pi]} \frac{2}{\pi}\frac{\theta}{1 - \cos(\theta)}. \tag{4.20}$$

Which is just a simple exercise to prove.

□

Remark 4.21. Denote the objective values

ω_r	–	value in the optimal solution of the relaxation (4.11)
ω^*	–	value in the optimal solution of the maxcut problem (4.4)
ω	–	value in the feasible solution of (4.4) obtained by the GW-algorithm

Then the following inequality holds trivially,

$$\omega \leq \omega^* \leq \omega. \tag{4.22}$$

The proposition 4.12 further implies

$$0.878\ \omega_r < E[\omega] \leq \omega^* \leq \omega_r. \tag{4.23}$$

Therefore, the bounds obtained by solving the relaxed problem are quite tight. What is more, the feasible solutions projected by GW-algorithm are also good in expectation. Generally, we need to solve the relaxation (4.11) only once and then repeat steps 2 and 3 of the algorithm couple of times. Keeping only the best solution one will (probably) get the objective at least as good as the expected value.

4.1.3 Semidefinite relaxation of the max-cut

Now the last arising question is how do we solve the relaxed problem (4.11)? Let us remind the formulation of the relaxed problem

$$\begin{array}{ll} \text{maximize} & \frac{1}{2}\sum_{i<j} w_{ij}(1 - v_i^T v_j), \\ \text{subject to} & v_i \in S^n \ (i = 1, \dots, n), \end{array} \tag{4.24}$$

where $v_i \in S^n$ are the variables, the scalar weights $w_{ij} \in \mathbb{R}_+$ are given and the indices i, j are in $\{1, \dots, n\}$.

Let $B := (v_1, \dots, v_n)$ be the $n \times n$ matrix with vectors v_i as columns. It holds that matrix $X := BB^T$ has elements $X_{ij} = v_i^T v_j$ for each $i, j \in (1, \dots, n)$, and X is positive semidefinite. Since $v_i \in S^n$ are unit vectors, the diagonal entries of X are ones, $X_{ii} = v_i^T v_i = 1$. We can find such X by solving following SDP and extract unit vectors v_i from Choelsky decomposition $X = BB^T$.

$$\begin{array}{ll} \text{maximize} & \frac{1}{4} L \bullet X, \\ \text{subject to} & diag(X) = e \\ & X \succeq 0, \end{array} \tag{4.25}$$

where $X \in \mathbb{S}^n_+$ is a variable, the matrix L is defined in (4.5), $diag(X)$ is the vector of diagonal entries of X, and e is the n-dimensional vector of ones.

Remark 4.26. In fact, we would receive the same semidefinite relaxation by following the general scheme for QCQP described in Section 3.1.

TODO: other combinatorial problems i.e. QAP, stable set, knapsack, ...

4.2 QAP

4.3 Stable set

4.4 Knapsack

Chapter 5

Computational comparison of the relaxation methods

A great comparison of various LP and SDP branch and bound algorithms for the minimum graph bisection problem was done in [23].

5.1 Branch and bound

We will first describe a general idea of the branch and bound algorithms. uppose we are dealing with minimization problem.

The basic idea of this method is to divide and conquer. There are 3 basics steps we will do repeatedly. Branch, compute bounds and prune.

- Branch. First divide by branching the original problem into the smaller subproblems i.e. by partitioning the feasible set. In combinatorial (discrete) problems this can be done by fixing a variable. For example if $x_l \in \{0, 1\}$, then by fixing $x_l = 1$ and $x_l = 0$ respectively, the problem is divided into two subproblems with one less free variable.

  Although, the branch and bound algorithms are mostly developed and used for solving combinatorial problems, it is also possible to adapt then for general QCQP. Whenever the feasible set is compact, the branching can be done by subdividing the feasible region into the Cartesian product of triangles and rectangles (see [21, 22]).

- Compute bounds. The lower bounds should be established for each branch. Usually this is done by solving the relaxed problem for each subproblem. An upper bound is computed on the optimal value, this can be done by finding a feasible solution for chosen subproblem. It is often extracted from the solution of the relaxed problem using a projection or rounding procedure.

- Prune. Having both lower and upper bounds, conquer by pruning all the branches with lower bound greater then global upper bound. The optimal solution is surely not in these branches.

- Repeat. This process is repeated by branching the remaining subproblems, computing the new (better) lower bounds for these smaller problems, improving the upper bound and further pruning the problem tree.

One can use more or less clever techniques to decide when, how and which subproblems to branch or how and when to compute bounds. Using stronger relaxations for bounding steps results in increased computation time, but hopefully less iterations are needed since better bounds allow to cut branches earlier. On the other hand weaker relaxations are computed faster so they let us do more iterations and explore much greater portion of the problem tree. The greater number of iterations increases chances for guessing the optimal solution as well as obtaining good lower bounds for problems that are small enough.

Appendix A

A.1 Schur complement

Theorem A.1 (Schur complement lemma for PSD). *Let M be a symmetric matrix of the form*

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

The following conditions are equivalent:

1. $M \succeq 0$ *(M is positive semidefinite).*
2. $A \succeq 0, \quad (I - AA^\dagger)B = 0, \quad C - B^T A^\dagger B \succeq 0.$
3. $A \succeq 0, \quad \mathcal{R}(B) \subseteq \mathcal{R}(A), \quad C - B^T A^\dagger B \succeq 0.$

(The roles of A and C can be switched.) Where $A^\dagger$ denotes pseudoinverse of A and $\mathcal{R}(B)$ denotes column range of B. [10, 5]

Proof. (will be added later)

A.2 Cones

Definition A.2 (Cone). Set $\mathcal{K}$ is cone if for all $x \in \mathcal{K}$ and all $\theta \geq 0$, it holds that $\theta x \in \mathcal{K}$.

Definition A.3 (Proper cone). The cone $\mathcal{K} \subseteq \mathbb{R}^n$ is a proper cone if it has following properties

- $\mathcal{K}$ is closed
- $\mathcal{K}$ is convex (for any $\theta_1, \theta_2 \geq 0$ and $x_1, x_2 \in \mathcal{K}$ also $\theta_1 x_1 + \theta_2 x_2 \in \mathcal{K}$)
- $\mathcal{K}$ has nonempty interior ($int\mathcal{K} \neq \emptyset$)
- $\mathcal{K}$ is pointed (does not contain whole line, i.e. if $\pm x \in \mathcal{K}$, then $x = 0$).

Definition A.4 (Dual cone)**.** For any cone $\mathcal{K}$ we define its dual cone $\mathcal{K}^*$ as a set

$$\mathcal{K}^* = \{z \mid \forall x \in \mathcal{K},\ x^T z \geq 0\}. \tag{A.5}$$

If $\mathcal{K} = \mathcal{K}^*$ we say cone $\mathcal{K}$ is selfdual.

TODO:

- carthesian product of proper/self-dual cones
- $\mathbb{R}^n_+$, Q_n, $\mathbb{S}^n_+$ are proper, self-dual cones + figures.
- example of converting SOCP to SDP
- example of converting QP to SOCP
- Moore-Penrose pseudoinverse

References

TODO: Unified refeference (same formats and orderings for source, year, pages, volume, journal names)

[1] S. Burer, S.Kim, M. Kojima, *Faster, but Weaker, Relaxations for Quadratically Constrained Quadratic Programs,* April, 2013
http://www.is.titech.ac.jp/~kojima/articles/socpsdp.pdf

[2] S. Kim, M. Kojima, *Exact Solutions of Some Nonconvex Quadratic Optimization Problems via SDP and SOCP Relaxations,*January 2002.Computational Optimization and Applications Vol.26 (2) 143-154 (2003)
http://www.is.titech.ac.jp/research/research-report/B/B-375.ps.gz

[3] S. Kim, M. Kojima, M. Yamashita, *Second Order Cone Programming Relaxation of a Positive Semidefinite Constraint,* July 2002. Optimization Methods and Software Vol.18 (5) 535-541 (2003).
http://www.is.titech.ac.jp/~kojima/articles/B-381.pdf

[4] S. Kim, M. Kojima, *Second Order Cone Programming Relaxations of Nonconvex Quadratic Optimization Problems,* Optim. Methods Softw., 15(3-4):201-224, 2001
http://www.researchgate.net/publication/2637515_Second_Order_Cone_Programming_Relaxation_of_Nonconvex_Quadratic_Optimization_Problems

[5] S. Boyd, L. Vandenberghe, *Convex Optimization,* Cambridge University Press 2004, seventh printing with corrections 2009
http://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf

[6] R. M. Freund, *Introduction to Semidefinite Programming (SDP),* 2009
http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-251j-introduction-to-mathematical-programming-fall-2009/readings/MIT6_251JF09_SDP.pdf

[7] F. Alizadeh, D. Goldfarb, *Second-order cone programming,* Springer, 2002

[8] K. Pokorná, *Second order cone programing, (Master's thesis),* 2013
http://www.iam.fmph.uniba.sk/studium/efm/diplomovky/2013/pokorna/diplomovka.pdf

[9] M. S. Lobo, L. Vandenberghe, S. Boyd, H. Lebret, *Applications of second-Order cone programming,* Linear Algebra and its Applications 284, 1998, 193-228
http://www.seas.ucla.edu/~vandenbe/publications/socp.pdf

[10] J. Gallier, *The Schur Complement and Symmetric Positive Semidefinite (and Definite) Matrices,* December 10, 2010
http://www.cis.upenn.edu/~jean/schur-comp.pdf

[11] R. Grone, C. Johnson, E. Sá, H. Wolkowicz, *Positive definite completions of partial hermitian matrices,* Linear Algebra Applications, 58:109-124, 1984.

[12] H. Wolkowicz, R. Saigal,L. Vandenberghe, *Handbook of Semidefinite Programming: Theory, Algorithms, and Applications,* ISBN 978-1-4615-4381-7, second printing 2003

[13] B. Kocuk, S. S. Dey, X. A. Sun, *Inexactness of SDP Relaxation and Valid Inequalities for Optimal Power Flow,* IEEE Transactions on Power Systems, 2014

[14] H. Dong, J. Linderoth, *On Valid Inequalities for Quadratic Programming with Continuous Variables and Binary Indicators* Springer Volume 7801 of the series Lecture Notes in Computer Science pp 169-180,

[15] M. X. Goemans, D. P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming* J. ACM, 42(6):1115-1145, 1995

[16] K. M. Anstreicher, *Semidefinite Programming versus the Reformulation Linearization Technique for Nonconvex Quadratically Constrained Quadratic Programming* Optimization Online, May 2007
http://www.optimization-online.org/DBHTML/2007/05/1655.html

[17] A. Qualizza , P. Belotti, F. Margot, *Linear Programming Relaxations of Quadratically Constrained Quadratic Programs* IMA Volume Series 2010
http://wpweb2.tepper.cmu.edu/fmargot/PDF/sdp.pdf

[18] H. D. Sherali, W. P. Adams, *A reformulation-linearization technique for solving discrete and continuous nonconvex problems* Kluwer, Dordrecht 1998

[19] H.D. Sherali, B.M.P. Fraticelli, *Enhancing RLT relaxations via a new class of semidefinite cuts* J. Global Optim. 22 (2002), 233-261

[20] X. Zheng, X. Sun, D. Li, *Nonconvex quadratically constrained quadratic programming: best d.c. decompositions and their sdp representations* Journal of Global Optimization, 50:695-712, 2011

[21] J. Linderoth, *A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs* Mathematical Programming, 103, 251–282 (2005)

[22] U. Raber, *A Simplicial Branch-and-Bound Method for Solving Nonconvex All-Quadratic Programs* Journal of Global Optimization 13: 417–432, 1998

[23] M. Armbruster, M. Fügenschuh, Ch. Helmberg, A. Martin, *LP and SDP branch-and-cut algorithms for the minimum graph bisection problem: a computational comparison* Math. Prog. Comp. (2012) 4:275–306

[24] P.M. Pardalos, S.A. Vavasis, *Quadratic programming with one negative eigenvalue is NP-hard* J. Global Optim. 1, 15–22, 1991