Semidefinite Optimization and Relaxation

Heng Yang

2024-03-04

Contents

Preface

This is the textbook for Harvard ENG-SCI 257: Semidefinite Optimization and Relaxation.

Feedback

I would like to invite you to provide comments to the textbook via the following two ways:

- Inline comments with Hypothesis:
  - Go to Hypothesis and create an account
  - Install the Chrome extension of Hypothesis
  - Provide public comments to textbook contents and I will try to address them
- Blog-style comments with Disqus:
  - At the end of each Chapter, there is a Disqus module where you can leave feedback

I would recommend using Disqus for high-level and general feedback regarding the entire Chapter, but using Hypothesis for feedback and questions about the technical details.

Offerings

Information about the offerings of the class is listed below.

2024 Spring

Time: Mon/Wed 2:15 - 3:30pm

Location: Science and Engineering Complex, 1.413

Instructor: Heng Yang

Teaching Fellow: Safwan Hossain

Syllabus

Notation

We will use the following standard notation throughout this book.

Basics

$\mathbb{R}$	real numbers
$\mathbb{R}_+$	nonnegative real
$\mathbb{R}_{++}$	positive real
$\mathbb{Z}$	integers
$\mathbb{N}$	nonnegative integers
$\mathbb{N}_+$	positive integers
$\mathbb{R}^n$	n-D column vector
$\mathbb{R}^n_+$	nonnegative orthant
$\mathbb{R}^n_{++}$	positive orthant
e_i	standard basic vector
$\Delta_n := \{x \in \mathbb{R}^n_+ \mid \sum x_i = 1\}$	standard simplex

Matrices

$\mathbb{R}^{m\times n}$	$m \times n$ real matrices
$\mathbb{S}^n$	$n \times n$ symmetric matrices
$\mathbb{S}^n_+$	$n \times n$ positive semidefinite matrices
$\mathbb{S}^n_{++}$	$n \times n$ positive definite matrices
$\langle A, B\rangle$ or $\bullet$	inner product in $\mathbb{R}^{m\times n}$
$\mathrm{tr}(A)$	trace of $A \in \mathbb{R}^{n\times n}$
$A^\top$	matrix transpose
$\det(A)$	matrix determinant
$\mathrm{rank}(A)$	rank of a matrix
$\mathrm{diag}(A)$	diagonal of a matrix A as a vector

$\mathrm{Diag}(a)$	turning a vector into a diagonal matrix
$\mathrm{BlkDiag}(A, B, \dots)$	block diagonal matrix with blocks $A, B, \dots$
$\succeq 0$ and $\preceq 0$	positive / negative semidefinite
$\succ 0$ and $\prec 0$	positive / negative definite
$\lambda_{\max}$ and $\lambda_{\min}$	maximum / minimum eigenvalue
$\sigma_{\max}$ and $\sigma_{\min}$	maximum / minimum singular value
$\mathrm{vec}(A)$	vectorization of $A \in \mathbb{R}^{m\times n}$
$\mathrm{svec}(A)$	symmetric vectorization of $A \in \mathbb{S}^n$
$\|A\|_{\mathrm{F}}$	Frobenius norm
$\mathrm{Range}(A)$	span of the column vectors
$\ker(A)$	right null space

Geometry

$\|a\|_p$	p-norm
$\|a\|$	2-norm
$B(o, r)$	ball with center o and radius r
$\mathrm{aff}(S)$	affine hull of set S
$\mathrm{conv}(S)$	convex hull of set S
$\mathrm{cone}(S)$	conical hull of set S
$\mathrm{int}(S)$	interior of set S
$\mathrm{ri}(S)$	relative interior of set S
∂S	boundary of set S
P°	polar of convex body
P^*	dual of set P
$\mathrm{O}(d)$	orthogonal group of dimension d
$\mathrm{SO}(d)$	special orthogonal group of dimension d
$\mathcal{S}^{d-1}$	unit sphere in $\mathbb{R}^d$

Optimization

KKT	Karush–Kuhn–Tucker

LP	linear program
QP	quadratic program
SOCP	second-order cone program
SDP	semidefinite program

Algebra

$\mathbb{R}[x]$	polynomial ring in x with real coefficients
deg	degree of a monomial / polynomial
$\mathbb{R}[x]_d$	polynomials in x of degree up to d
$[x]_d$	vector of monomials of degree up to d
$[\![x]\!]_d$	vector of monomials of degree d

Chapter 1

Mathematical Background

1.1 Convexity

A very important notion in modern optimization is that of *convexity*. To a large extent, an optimization problem is "easy" if it is convex, and "difficult" when convexity is lost, i.e., *nonconvex*. We give a basic review of convexity here and refer the reader to (Rockafellar, 1970), (Boyd and Vandenberghe, 2004), and (Bertsekas et al., 2003) for comprehensive treatments.

We will work on a finite-dimensional real vector space, which we will identify with $\mathbb{R}^n$.

Definition 1.1 (Convex Set)**.** A set S is convex if $x_1, x_2 \in S$ implies $\lambda x_1 + (1-\lambda)x_2 \in S$ for any $\lambda \in [0,1]$. In other words, if $x_1, x_2 \in S$, then the line segment connecting x_1 and x_2 lies inside S.

Conversely, a set S is nonconvex if Definition 1.1 does not hold.

Given $x_1, x_2 \in S$, $\lambda x_1 + (1-\lambda)x_2$ is called a *convex combination* when $\lambda \in [0,1]$. For convenience, we will use the following notation

$$\begin{aligned}(x_1, x_2) &= \{\lambda x_1 + (1-\lambda)x_2 \mid \lambda \in (0,1)\},\\ [x_1, x_2] &= \{\lambda x_1 + (1-\lambda)x_2 \mid \lambda \in [0,1]\}.\end{aligned} \tag{1.1}$$

A **hyperplane** is a common convex set defined as

$$H = \{x \in \mathbb{R}^n \mid \langle c, x\rangle = d\} \tag{1.2}$$

for some $c \in \mathbb{R}^n$ and scalar d. A **halfspace** is a convex set defined as

$$H^+ = \{x \in \mathbb{R}^n \mid \langle c, x\rangle \geq d\}. \tag{1.3}$$

Given two nonempty convex sets C_1 and C_2, the **distance** between C_1 and C_2 is defined as

$$\operatorname{dist}(C_1, C_2) = \inf\{\|c_1 - c_2\| \mid c_1 \in C_1, c_2 \in C_2\}. \tag{1.4}$$

For a convex set C, the hyperplane H in (1.2) is called a **supporting hyperplane** for C if C is contained in the half space H^+ and the distance between H and C is zero. For example, the hyperplane $x_1 = 0$ is supporting for the hyperboloid $\{(x_1, x_2) \mid x_1 x_2 \geq 1, x_1 \geq 0, x_2 \geq 0\}$ in $\mathbb{R}^2$.

An important property of a convex set is that we can *certify* when a point is not in the set. This is usually done via a separation theorem.

Theorem 1.1 (Separation Theorem)**.** *Let S_1, S_2 be two convex sets in $\mathbb{R}^n$ and $S_1 \cap S_2 = \emptyset$, then there exists a hyperplane that separates S_1 and S_2, i.e., there exists c and d such that*

$$\begin{aligned} \langle c, x\rangle \geq d, &\forall x \in S_1, \\ \langle c, x\rangle \leq d, &\forall x \in S_2. \end{aligned} \tag{1.5}$$

Further, if S_1 is compact (i.e., closed and bounded) and S_2 is closed, then the separation is strict, i.e., the inequalities in (1.5) *are strict.*

The strict separation theorem is used typically when S_1 is a single point (hence compact).

We will see a generalization of the separation theorem for nonconvex sets later after we introduce the idea of sums of squares.

Exercise 1.1. Provide examples of two disjoint convex sets such that the separation in (1.5) is not strict in one way and both ways.

Exercise 1.2. Provide a constructive proof that the separation hyperplane exists in Theorem 1.1 when (1) both S_1 and S_2 are closed, and (2) at least one of them is bounded.

The intersection of convex sets is always convex (try to prove this).

1.2 Convex Geometry

1.2.1 Basic Facts

Given a set S, its **affine hull** is the set

$$\operatorname{aff}(S) = \left\{ \sum_{i=1}^{k} \lambda_i u_i \mid \lambda_1 + \cdots + \lambda_k = 1, u_i \in S, k \in \mathbb{N}_+ \right\},$$

where $\sum_{i=1}^{k} \lambda_i u_i$ is called an *affine combination* of $u_1, \ldots, u_k$ when $\sum_i \lambda_i = 1$. The affine hull of a set is the smallest affine subspace that contains S, and the **dimension** of S is the dimension of its affine hull. The affine hull of the emptyset is the emptyset, of a singleton is the singleton itself. The affine hull of a set of two different points is the line going through them. The affine hull of a set of three points not on one line is the plane going through them. The affine hull of a set of four points not in a plane in $\mathbb{R}^3$ is the entire space $\mathbb{R}^3$.

For a convex set $C \subseteq \mathbb{R}^n$, the **interior** of C is defined as

$$\text{int}(C) := \{u \in C \mid \exists \epsilon > 0, B(u, \epsilon) \subseteq C\},$$

where $B(u, \epsilon)$ denotes a ball centered at u with radius ϵ (using the usual 2-norm). Each point in $\text{int}(C)$ is called an *interior point* of C. If $\text{int}(C) = C$, then C is said to be an **open set**. A convex set with nonempty interior is called a **convex domain**, while a compact (i.e., closed and bounded) convex domain is called a **convex body**.

The **boundary of** C is the subset of points that are in the **closure**[1] of C but are not in the interior of C, and we denote it as ∂C. For example, the closed line segment $C = [0, 1]$ has two points on the boundary: 0 and 1; the open line segment $C = (0, 1)$ has the same two points as its boundary.

It is possible that a convex set has empty interior. For example, a hyperplane has no interior, and neither does a singleton. In such cases, the **relative interior** can be defined as

$$\text{ri}(C) := \{u \in C \mid \exists \epsilon > 0, B(u, \epsilon) \cap \text{aff}(C) \subseteq C\}.$$

For a nonempty convex set, the relative interior always exists. If $\text{ri}(C) = C$, then C is said to be **relatively open**. For example, the relative interior of a singleton is the singleton itself, and hence a singleton is relatively open.

For a convex set C, a point $u \in C$ is called an **extreme point** if

$$u \in (x, y), x \in C, y \in C \quad \Rightarrow u = x = y.$$

For example, consider $C = \{(x, y) \mid x^2 + y^2 \leq 1\}$, then all the points on the boundary $\partial C = \{(x, y) \mid x^2 + y^2 = 1\}$ are extreme points.

A subset $F \subseteq C$ is called a **face** if F itself is convex and

$$u \in (x, y), u \in F, x, y \in C \quad \Rightarrow x, y \in F.$$

Clearly, the empty set $\emptyset$ and the entire set C are faces of C, which are called *trivial faces*. The face F is said to be *proper* if $F \neq C$. The set of any single

[1] The closure of a subset C of points, denoted $\text{cl}(C)$, consists of all points in C together with all limit points of C. The closure of C may equivalently be defined as the intersection of all closed sets containing C. Intuitively, the closure can be thought of as all the points that are either in C or "very near" C. For example, the closure of the open line segment $C = (0, 1)$ is the closed line segment $C = [0, 1]$.

extreme point is also a face. A face F of C is called **exposed** if there exists a supporting hyperplane H for C such that

$$F = H \cap C.$$

1.2.2 Cones, Duality, Polarity

Definition 1.2 (Polar)**.** For a nonempty set $T \subseteq \mathbb{R}^n$, its polar is the set

$$T^\circ := \{y \in \mathbb{R}^n \mid \langle x, y\rangle \leq 1, \forall x \in T\}. \tag{1.6}$$

The polar T° is a closed convex set and contains the origin. Note that T is always contained in the polar of T°, i.e., $T \subseteq (T^\circ)^\circ$. Indeed, they are equal under some assumptions.

Theorem 1.2 (Bipolar)**.** *If $T \subseteq \mathbb{R}^n$ is a closed convex set containing the origin, then $(T^\circ)^\circ = T$.*

An important class of convex sets are those that are invariant under positive scalings.[2] A set $K \subseteq \mathbb{R}^n$ is a **cone** if $tx \in K$ for all $x \in K$ and for all $t > 0$. For example, the positive real line $\{x \in \mathbb{R} \mid x > 0\}$ is a cone. The cone K is **pointed** if $K \cap -K = \{0\}$. It is said to be **solid** if its interior $\text{int}(K) \neq \emptyset$. Any nonzero point of a cone cannot be extreme. If a cone is pointed, the only extreme point is the origin.

The analogue of extreme point for convex cones is the **extreme ray**. For a convex cone K and $0 \neq u \in K$, the line segment

$$u \cdot [0, \infty) := \{tu \mid t \geq 0\}$$

is called an extreme ray of K if

$$u \in (x, y), x, y \in K \quad \Rightarrow \quad u, x, y \text{ are parallel to each other.}$$

If $u \cdot [0, \infty)$ is an extreme ray, then we say u generates the extreme ray.

Definition 1.3 (Proper Cone)**.** A cone K is proper if it is closed, convex, pointed, and solid.

A proper cone K induces a **partial order** on the vector space, via $x \succeq y$ if $x - y \in K$. We also use $x \succ y$ if $x - y$ is in $\text{int}(K)$. Important examples of proper cones are the nonnegative orthant, the second-order cone, the set of symmetric positive semidefinite matrices, and the set of nonnegative polynomials, which we will describe later in the book.

[2] Some authors define a cone using nonnegative scalings.

Definition 1.4 (Dual)**.** The dual of a nonempty set S is

$$S^* := \{y \in \mathbb{R}^n \mid \langle y, x\rangle \geq 0, \forall x \in S\}.$$

Given any set S, its dual S^* is always a closed convex cone. Duality reverses inclusion, that is,

$$S_1 \subseteq S_2 \quad \Rightarrow \quad S_1^* \supseteq S_2^*.$$

If S is a closed convex cone, then $S^{**} = S$. Otherwise, S^{**} is the closure of the smallest convex cone that contains S.

For a cone $K \subseteq \mathbb{R}^n$, one can show that

$$K^\circ = \{y \in \mathbb{R}^n \mid \langle x, y\rangle \leq 0, \forall x \in K\}.$$

The set K° is called the **polar cone** of K. The negative of K° is just the **dual cone**

$$K^* = \{y \in \mathbb{R}^n \mid \langle x, y\rangle \geq 0, \forall x \in K\}.$$

Definition 1.5 (Self-dual)**.** A cone K is self-dual if $K^* = K$.

As an easy example, the nonnegative orthant $\mathbb{R}^n_+$ is self-dual.

Example 1.1 (Second-order Cone)**.** The second-order cone, or the Lorentz cone, or the ice cream cone

$$\mathcal{Q}_n := \{(x_0, x_1, \dots, x_n) \in \mathbb{R}^{n+1} \mid \sqrt{x_1^2 + \cdots + x_n^2} \leq x_0\}$$

is a proper cone of $\mathbb{R}^{n+1}$. We will show that it is also self-dual.

Proof. Consider $(y_0, y_1, \dots, y_n) \in \mathcal{Q}_n$, we want to show that

$$x_0y_0 + x_1y_1 + \cdots + x_ny_n \geq 0, \forall (x_0, x_1, \dots, x_n) \in \mathcal{Q}_n. \tag{1.7}$$

This is easy to verify because

$$x_1y_1 + \cdots + x_ny_n \geq -\sqrt{x_1^2 + \cdots + x_n^2}\sqrt{y_1^2 + \cdots + y_n^2} \geq -x_0y_0.$$

Hence we have $\mathcal{Q}_n \subseteq \mathcal{Q}_n^*$.

Conversely, if (1.7) holds, then take

$$x_1 = -y_1, \dots, x_n = -y_n, \quad x_0 = \sqrt{x_1^2 + \cdots + x_n^2},$$

we have

$$y_0 \geq \sqrt{y_1^2 + \cdots + y_n^2},$$

hence $\mathcal{Q}_n^* \subseteq \mathcal{Q}_n$. ■

Not every proper cone is self-dual.

Exercise 1.3. Consider the following proper cone in $\mathbb{R}^2$

$$K = \{(x_1, x_2) \mid 2x_1 - x_2 \geq 0, 2x_2 - x_1 \geq 0\}.$$

Show that it is not self-dual.

1.3 Convex Optimization

Definition 1.6 (Convex Function)**.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function if

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y), \forall \lambda \in [0,1], \forall x, y \in \mathbb{R}^n.$$

A function f is convex if and only if its **epigraph** $\{(x,t) \in \mathbb{R}^{n+1} \mid f(x) \leq t\}$ is a convex set.

When a function f is differentiable, then there are several equivalent characterizations of convexity, in terms of the gradient $\nabla f(x)$ or the Hessian $\nabla^2 f(x)$.

Theorem 1.3 (Equivalent Characterizations of Convexity)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a twice differentiable function. The following propositions are equivalent.*

i. f is convex, i.e.,

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y), \forall \lambda \in [0,1], x, y \in \mathbb{R}^n.$$

ii. The first-order convexity condition holds:

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle, \forall x, y \in \mathbb{R}^n,$$

i.e., the hyperplane going through $(x, f(x))$ with slope $\nabla f(x)$ supports the epigraph of f.

iii. The second-order convexity condition holds:

$$\nabla^2 f(x) \succeq 0, \forall x \in \mathbb{R}^n,$$

i.e., the Hessian is positive semidefinite everywhere.

Let's work on a little exercise.

Exercise 1.4. Which one of the following functions $f : \mathbb{R}^n \to \mathbb{R}$ is not convex?

a. $\exp(-c^\top x)$, with c constant

b. $\exp(c^\top x)$, with c constant

c. $\exp(x^\top x)$

d. $\exp(-x^\top x)$

1.3.1 Minimax Theorem

Given a function $f : X \times Y \to \mathbb{R}$, the following inequality always holds

$$\max_{y\in Y} \min_{x\in X} f(x,y) \le \min_{x\in X} \max_{y\in Y} f(x,y). \tag{1.8}$$

If the maximum or minimum is not attained, then (1.8) holds with max / min replaced by sup and inf, respectively.

Exercise 1.5. Provide examples of f such that the inequality in (1.8) is strict.

It is of interest to understand when equality holds in (1.8).

Theorem 1.4 (Minimax Theorem)**.** *Let $X \subset \mathbb{R}^n$ and $Y \subset \mathbb{R}^n$ be compact convex sets, and $f : X \times Y \to \mathbb{R}$ be a continuous function that is convex in its first argument and concave in the second. Then*

$$\max_{y\in Y} \min_{x\in X} f(x,y) = \min_{x\in X} \max_{y\in Y} f(x,y).$$

A special case of this theorem, used in game theory to prove the existence of equilibria for zero-sum games, is when X and Y are standard unit simplicies and the function $f(x,y)$ is bilinear. In a research from our group (Tang et al., 2023), we used the minimax theorem to convert a minimax problem into a single-level minimization problem.

1.3.2 Lagrangian Duality

Consider a nonlinear optimization problem

$$\begin{aligned} u^\star = \min_{x\in\mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \le 0, i = 1, \dots, m, \\ & h_j(x) = 0, j = 1, \dots, p. \end{aligned} \tag{1.9}$$

Define the **Lagrangian** associated with the optimization problem (1.9) as

$$\begin{aligned} L : \mathbb{R}^n \times \mathbb{R}^m_+ \times \mathbb{R}^p \quad &\to \quad \mathbb{R}, \\ (x,\lambda,\mu) \quad &\mapsto \quad f(x) + \sum_{i=1}^m \lambda_i g_i(x) + \sum_{j=1}^p \mu_j h_j(x). \end{aligned} \tag{1.10}$$

The **Lagrangian dual function** is defined as

$$\phi(\lambda,\mu) := \min_{x\in\mathbb{R}^n} L(x,\lambda,\mu). \tag{1.11}$$

Maximizing this function over the dual variables (λ,μ) yields

$$v^\star := \max_{\lambda\ge 0, \mu\in\mathbb{R}^p} \phi(\lambda,\mu) \tag{1.12}$$

Applying the minimax Theorem 1.4, we can see that

$$v^\star = \max_{(\lambda,\mu)} \min_x L(x,\lambda,\mu) \leq \min_x \max_{(\lambda,\mu)} L(x,\lambda,\mu) = u^\star.$$

That is to say solving the dual problem (1.12) always provides a lower bound to the primal problem (1.9).

If the functions f, g_i are convex and h_i are affine, the Lagrangian is convex in x and convex in (λ, μ). To ensure strong duality (i.e., $u^\star = v^\star$), compactness or other **constraint qualifications** are needed. An often used condition is the Slater constraint qualification.

Definition 1.7 (Slater Constraint Qualification)**.** There exists a strictly feasible point for (1.9), i.e., a point $z \in \mathbb{R}^n$ such that $h_j(z) = 0, j = 1, \dots, p$ and $g_i(z) < 0, i = 1, \dots, m$.

Under these conditions, we have strong duality.

Theorem 1.5 (Strong Duality)**.** *Consider the optimization* (1.9) *and assume f, g_i are convex and h_j are affine. If Slater's constraint qualification holds, then the optimal value of the primal problem* (1.9) *is the same as the optimal value of the dual problem* (1.12)*.*

1.3.3 KKT Optimality Conditions

Consider the nonlinear optimization problem (1.9). A pair of primal and dual variables $(x^\star, \lambda^\star, \mu^\star)$ is said to satisfy the Karush-Kuhn-Tucker (KKT) optimality conditions if

$$\begin{aligned}
\text{primal feasibility}: &\quad g_i(x^\star) \leq 0, \forall i = 1, \dots, m; h_j(x^\star) = 0, \forall j = 1, \dots, p \\
\text{dual feasibility}: &\quad \lambda_i^\star \geq 0, \forall i = 1, \dots, m \\
\text{stationarity}: &\quad \nabla_x L(x^\star, \lambda^\star, \mu^\star) = 0 \\
\text{complementarity}: &\quad \lambda_i^\star \cdot g_i(x^\star) = 0, \forall i = 1, \dots, m.
\end{aligned} \tag{1.13}$$

Under certain constraint qualifications, the KKT conditions are necessary for local optimality.

Theorem 1.6 (Necessary Optimality Conditions)**.** *Assume any of the following constraint qualifications hold:*

- *The gradients of the constraints $\{\nabla g_i(x^\star)\}_{i=1}^m$, $\{\nabla h_j(x^\star)\}_{j=1}^p$ are linearly independent.*
- *Slater's constraint qualification (cf. Definition 1.7).*

- *All constraints $g_i(x)$ and $h_j(x)$ are affine functions.*

Then, at every local minimum $x^\star$ of (1.9)*, the KKT conditions* (1.13) *hold.*

On the other hand, for convex optimization problems, the KKT conditions are sufficient for global optimality.

Theorem 1.7 (Sufficient Optimality Conditions)**.** *Assume optimization* (1.9) *is convex, i.e., f, g_i are convex and h_j are affine. Every point $x^\star$ that satisfies the KKT conditions* (1.13) *is a global minimizer.*

1.4 Linear Optimization

1.4.1 Polyhedra

In $\mathbb{R}^n$, a **polyhedron** is a set defined by finitely many linear inequalities, i.e.,

$$P = \{x \in \mathbb{R}^n \mid Ax \geq b\}, \tag{1.14}$$

for some matrix $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. In (1.14), the inequality should be interpreted as $Ax - b \in \mathbb{R}^m_+$, i.e., every entry of Ax is no smaller than the corresponding entry of b.

The convex hull of finitely many points in $\mathbb{R}^n$ is called a **polytope**, where the convex hull of a set S is defined as

$$\operatorname{conv}(S) = \left\{ \sum_{i=1}^{k} \lambda_i u_i \mid k \in \mathbb{N}_+, \sum_{i=1}^{k} \lambda_i = 1, \lambda_i \geq 0, i = 1, \dots, k, u_i \in S, \forall i = 1, \dots, k \right\}, \tag{1.15}$$

i.e., all possible convex combinations of points in S. Clearly, a polytope is bounded.

The conic hull of finitely many points in $\mathbb{R}^n$ is called a **polyhedral cone**, where the conic hull of a set S is defined as

$$\operatorname{cone}(S) = \left\{ \sum_{i=1}^{k} \lambda_i u_i \mid k \in \mathbb{N}_+, \lambda_i \geq 0, i = 1, \dots, k, u_i \in S, \forall i = 1, \dots, k \right\}. \tag{1.16}$$

The only difference between (1.16) and (1.15) is the removal of $\sum_i \lambda_i = 1$. Clearly, the origin belongs to the conic hull of any nonempty set, and the conic hull of any nonempty set is unbounded.

The next theorem characterizes a polyhedron.

Theorem 1.8 (Polyhedron Decomposition)**.** *Every polyhedron P is finitely generated, i.e., it can be written as the Minkowski sum of a polytope and a polyhedral cone:*

$$P = \text{conv}(u_1, \dots, u_r) + \text{cone}(v_1, \dots, v_s),$$

where the Minkowski sum of two sets is defined as $X + Y := \{x + y \mid x \in X, y \in Y\}$.

Further, a bounded polyhedron is a polytope.

An extreme point of a polytope is called a **vertex**. A 1-dimensional face of a polytope is called an **edge**. A $d-1$-dimensional face of a d-dimensional polytope is called a **facet**.

1.4.2 Linear Program

We will now give a brief review of important results in linear programming (LP). The standard reference for linear programming is (Bertsimas and Tsitsiklis, 1997). In some sense, the theory of semidefinite programming (SDP) has been developed in order to generalize those of LP to the setup where the decision variable becomes a symmetric matrix and the inequality is interpreted as being positive semidefinite.

A standard form linear program (LP) reads

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \langle c, x \rangle \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{1.17}$$

for given $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$. Often the tuple (A, b, c) is called the *problem data* because the LP (1.17) is fully defined once the tuple is given (indeed many LP numerical solvers take the tuple (A, b, c) as input). Clearly, the feasible set of the LP (1.17) is a polyhedron. The LP (1.17) is often referred to as the **primal** LP. Associated with (1.17) is the following **dual** LP

$$\begin{aligned} \max_{y \in \mathbb{R}^m} \quad & \langle b, y \rangle \\ \text{s.t.} \quad & c - A^\top y \geq 0 \end{aligned} \tag{1.18}$$

It is worth noting that the dimension of the dual variable y is exactly the number of constraints in the primal LP.

Lagrangian duality. Let us use the idea of Lagrangian duality introduced in Section 1.3.2 to verify that (1.18) is indeed the Lagrangian dual problem of (1.17). The Lagrangian associated with (1.17) is

$$\begin{aligned} L(x, \lambda, \mu) &= \langle c, x \rangle + \langle \mu, Ax - b \rangle + \langle \lambda, -x \rangle, \quad \mu \in \mathbb{R}^m, \lambda \in \mathbb{R}^n_+ \\ &= \langle c + A^\top \mu - \lambda, x \rangle - \langle \mu, b \rangle, \quad \mu \in \mathbb{R}^m, \lambda \in \mathbb{R}^n_+. \end{aligned} \tag{1.19}$$

The Lagrangian dual function is therefore

$$\phi(\lambda,\mu)=\min_x L(x,\lambda,\mu)=\begin{cases}-\langle\mu,b\rangle & \text{if } c+A^\top\mu-\lambda=0\\ -\infty & \text{Otherwise}\end{cases},\mu\in\mathbb{R}^m,\lambda\in\mathbb{R}^n_+.$$

The Lagrangian dual problem seeks to maximize the dual function $\phi(\lambda,\mu)$, and hence it must set $c+A^\top\mu-\lambda=0$ (otherwise it leads to $-\infty$). As a result, the dual problem is

$$\begin{aligned}\max_{\mu\in\mathbb{R}^m}\quad & \langle b,-\mu\rangle\\ \text{s.t.}\quad & c+A^\top\mu=\lambda\geq 0\end{aligned}\tag{1.20}$$

With a change of variable $y:=-\mu$, we observe that problem (1.20) is precisely problem (1.18).

Weak duality. For the pair of primal-dual LPs, it is easy to verify that, for any x that is feasible for the primal (1.17) and y that is feasible for the dual (1.18), we have

$$\langle c,x\rangle-\langle b,y\rangle=\langle c,x\rangle-\langle Ax,y\rangle=\langle c,x\rangle-\langle A^\top y,x\rangle=\langle c-A^\top y,x\rangle\geq 0.\tag{1.21}$$

Therefore, denoting $p^\star$ as the optimum of (1.17) and $d^\star$ as the optimum of (1.18), we have the weak duality

$$p^\star\geq d^\star.$$

Note that such weak duality can also be directly obtained since (1.20) is the Lagrangian dual of (1.17).

If $p^\star=d^\star$, then we say **strong duality** holds. The LP (1.17) is said to be **feasible** if its feasible set is nonempty. It is said to be **unbounded below** if there exists a sequence $\{u_i\}_{i=1}^\infty\subseteq\mathbb{R}^n_+$ such that $\langle c,u_i\rangle\to-\infty$ and $Au_i=b$. If the primal (1.17) is infeasible (resp. unbounded below), we set $p^\star=+\infty$ (resp. $p^\star=-\infty$). Similar characteristics are defined for the dual LP (1.18). In particular, if the dual (1.18) is unbounded, then we set $d^\star=+\infty$. If the dual is infeasible, then we set $d^\star=-\infty$.

Strong duality is well understood in linear programming.

Theorem 1.9 (LP Strong Duality)**.** *For the LP primal-dual pair* (1.17) *and* (1.18)*, we have*

- *If one of* (1.17) *and* (1.18) *is feasible, then $p^\star=d^\star$ (i.e., finite, $+\infty$, or $-\infty$).*
- *If one of $p^\star$ or $d^\star$ is finite, then $p^\star=d^\star$ is finite, and both* (1.17) *and* (1.18) *achieve the same optimal value (i.e., they botb have optimizers).*

- *A primal feasible point $x^\star$ of* (1.17) *is a minimizer if and only if there exists a dual feasible point $y^\star$ such that $\langle c, x^\star\rangle = \langle b, y^\star\rangle$.*

For example, consider the following primal-dual LP pair

$$\begin{cases} \min_{x\in\mathbb{R}^3_+} & x_1 + x_2 + 2x_3 \\ \text{s.t.} & \begin{bmatrix} -1 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{cases}, \begin{cases} \max_{y\in\mathbb{R}^2} & y_2 \\ \text{s.t.} & \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} - \begin{bmatrix} -1 & 1 \\ 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \geq 0 \end{cases}. \tag{1.22}$$

$x^\star = [1/2, 1/2, 0]^\top$ is feasible for the primal and attains $p^\star = 1$. $y^\star = [0, 1]^\top$ is feasible for the dual and attains $d^\star = 1$. Therefore, both $x^\star$ and $y^\star$ are optimizers for the primal and dual, respectively.

Complementary slackness. Strong duality, when combined with (1.21), implies that

$$x_i^\star (c - A^\top y^\star)_i = 0, \forall i = 1, \dots, n,$$

where $(\cdot)_i$ denotes the i-th entry of a vector. This is known as complementary slackness, which states that whenever a primal optimal solution has a nonzero entry, the corresponding dual inequality must be tight.

An important property of LP is that if the primal problem is feasible and bounded below, then it must have an optimizer that is a **basic feasible point**, i.e., a feasible point has at most m nonzero entries. The simplex method (Bertsimas and Tsitsiklis, 1997) for solving LPs searches for optimizers among the basic feasible points.

We also introduce how to detect infeasibility and unboundedness of LPs.

Theorem 1.10 (LP Infeasibility and Unboundedness)**.** *Infeasibility and Unboundedness of LP can be certified by existence of an improving/decreasing ray for the primal and dual:*

- *When the primal* (1.17) *is feasible, it is unbounded below if and only if it has a decreasing ray, i.e., there exists $u \in \mathbb{R}^n$ such that*

$$Au = 0, \quad u \geq 0, \quad \langle c, u\rangle < 0.$$

- *When the dual* (1.18) *is feasible, it is unbounded above if and only if it has an improving ray, i.e., there exists $u \in \mathbb{R}^m$ such that*

$$A^\top u \leq 0, \quad \langle b, u\rangle > 0.$$

- *The primal problem* (1.17) *is infeasible if and only if the dual problem* (1.18) *has an improving ray, i.e., there exists $u \in \mathbb{R}^m$ such that*

$$A^\top u \leq 0, \quad \langle b, u\rangle > 0.$$

- *The dual problem* (1.18) *is infeasible if and only if the primal problem* (1.17) *has a decreasing ray, i.e., there exists* $u \in \mathbb{R}^n$ *such that*

$$Au = 0, \quad u \geq 0, \quad \langle c, u \rangle < 0.$$

It is important to note that both the primal and dual can be infeasible, as in the following example.

$$\begin{cases} \min_{x \in \mathbb{R}^2_+} & -x_1 - x_2 \\ \text{s.t.} & \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} x = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \end{cases}, \begin{cases} \max_{y \in \mathbb{R}^2} & 2y_1 + 3y_2 \\ \text{s.t.} & \begin{bmatrix} -1 \\ -1 \end{bmatrix} - \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} y \geq 0 \end{cases}. \tag{1.23}$$

1.4.3 Farkas Lemma

A foundational result in linear programming is the Farkas Lemma.

Theorem 1.11 (Farkas Lemma)**.** *For a given* $A \in \mathbb{R}^{m \times n}$ *and* $c \in \mathbb{R}^n$*, if* $\langle c, x \rangle \geq 0$ *for all* x *satisfying* $Ax \geq 0$*, then there exists* $\lambda \in \mathbb{R}^m$ *such that*

$$c = A^\top \lambda, \quad \lambda \geq 0.$$

As a simple example, take $A = \mathrm{I}_n$ as the identity matrix, then Farkas Lemma says if $\langle c, x \rangle \geq 0$ for all $x \geq 0$, then c must be that $c \geq 0$ – this is exactly the fact that the nonnegative orthant $\mathbb{R}^n_+$ is self-dual.

In general, the Farkas Lemma states if the linear function $\langle c, x \rangle$ is nonnegative on the space $\{Ax \geq 0\}$, then there exists $\lambda \in \mathbb{R}^m_+$ such that

$$\langle c, x \rangle = \langle \lambda, Ax \rangle = \sum_{i=1}^m \lambda_i (a_i^\top x), \tag{1.24}$$

where $a_i^\top$ is the i-th row of A. Note that (1.24) is a polynomial identity. As we will see later in the course, the idea of sums of squares (SOS), to some extent, is to generalize Farkas Lemma to the case where the function is a polynomial and the set is a basic semialgebraic set (i.e., defined by polynomial equalities and inequalities).

A generalization of Farkas Lemma to inhomogeneous affine functions is stated below.

Theorem 1.12 (Inhomogeneous Farkas Lemma)**.** *Suppose the set* $P = \{x \in \mathbb{R}^n \mid Ax \geq b\}$ *with* $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$ *is nonempty. If a linear function* $\langle c, x \rangle - d$ *is nonnegative on* P*, then there exists* $\lambda \in \mathbb{R}^m$ *and* $\nu \in \mathbb{R}$ *such that*

$$\langle c, x \rangle - d = \nu + \langle \lambda, Ax - b \rangle, \quad \lambda \geq 0, \nu \geq 0.$$

A more general result is called the Theorem of Alternatives, which states that a polyhedral set is empty if and only if another polyhedral set is nonempty.

Theorem 1.13 (Theorem of Alternatives)**.** *Given* $A_1 \in \mathbb{R}^{m_1 \times n}, A_2 \in \mathbb{R}^{m_2 \times n}$, $b_1 \in \mathbb{R}^{m_1}$, *and* $b_2 \in \mathbb{R}^{m_2}$, *the set*

$$\{x \in \mathbb{R}^n \mid A_1 x > b_1, A_2 x \geq b_2\}$$

is empty if and only if the following set

$$\left\{ (\lambda_1, \lambda_2) \in \mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \;\middle|\; \begin{array}{r} \lambda_1 \geq 0, \lambda_2 \geq 0, \\ b_1^\top \lambda_1 + b_2^\top \lambda_2 \geq 0, \\ A_1^\top \lambda_1 + A_2^\top \lambda_2 = 0, \\ (e + b_1)^\top \lambda_1 + b_2^\top \lambda_2 = 1 \end{array} \right\}$$

is nonempty, with e *being the vector of all ones.*

Chapter 2

Semidefinite Optimization

2.1 Positive Semidefinite Matrices

A real matrix $A = (A_{ij}) \in \mathbb{R}^{n\times n}$ is symmetric if $A = A^\top$, i.e., $A_{ij} = A_{ji}$ for all i, j. Let $\mathbb{S}^n$ be the space of all real symmetric matrices.

Any symmetric matrix A defines a **quadratic form** $x^\top Ax$. A matrix A is said to be **positive semidefinite** (PSD) if and only if its associated quadratic form is nonnegative, i.e.,

$$x^\top Ax \geq 0, \quad \forall x \in \mathbb{R}^n.$$

We use $\mathbb{S}^n_+$ to denote the set of $n \times n$ PSD matrices. We also write $A \succeq 0$ to denote positive semidefiniteness when the dimension is clear.

There are several equivalent characterizations of positive semidefiniteness.

Lemma 2.1 (Positive Semidefinite Matrices)**.** *Let $A \in \mathbb{S}^n$ be a symmetric matrix, the following statements are equivalent:*

1. *A is positive semidefinite.*
2. *$x^\top Ax \geq 0, \forall x \in \mathbb{R}^n$.*
3. *All eigenvalues of A are nonnegative.*
4. *All $2^n - 1$ principal minors of A are nonnegative.*
5. *The coefficients of $p_A(\lambda)$ weakly alternate in sign, i.e., $(-1)^{n-k}p_k \geq 0$ for $k = 0, \ldots, n-1$, where $p_A(\lambda) = \det(A - \lambda \mathrm{I}_n)$ is the characteristics polynomial of A.*
6. *There exists a factorization $A = BB^\top$, where $B \in \mathbb{R}^{n\times r}$ with r the rank of A.*

Among the equivalent characterizations of PSD matrices, (5) is less well-known, but it can be very useful when we want to convert a PSD constraint into multiple scalar constraints. For example, consider the following subset of $\mathbb{R}^3$:

$$\left\{ z \in \mathbb{R}^3 \;\middle|\; X(z) = \begin{bmatrix} 1 & z_1 & z_2 \\ z_1 & z_2 & z_3 \\ z_2 & z_3 & 5z_2 - 4 \end{bmatrix} \succeq 0 \right\}.$$

We can first form the characteristic polynomial of $X(z)$ –whose coefficients will be functions of z– and then invoking (5) to obtain a finite number scalar inequality constraints. We can then pass these scalar constraints to Mathematica and plot the set as in the following figure (Yang et al., 2022).

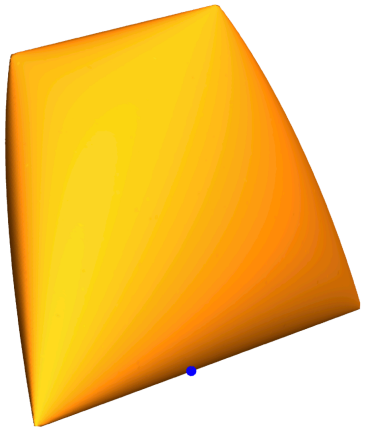

Figure 2.1: An example spectrahedron.

Similarly, we say a matrix $A \in \mathbb{S}^n$ is **positive definite** (PD) is its associated quadratic form is always positive, i.e.,

$$x^\top A x > 0, \quad \forall x \in \mathbb{R}^n.$$

We use $\mathbb{S}^n_{++}$ to denote the set of $n \times n$ PD matrices, and also write $A \succ 0$ when the dimension is clear.

Below is set of equivalent characterizations of positive definite matrices.

Lemma 2.2 (Positive Definite Matrices)**.** *Let $A \in \mathbb{S}^n$ be a symmetric matrix, the following statements are equivalent:*

1. *A is positive definite.*
2. *$x^\top A x > 0, \forall x \in \mathbb{R}^n$.*
3. *All eigenvalues of A are strictly positive.*

4. *All n leading principal minors of A are strictly positive.*

5. *The coefficients of $p_A(\lambda)$ strictly alternate in sign, i.e., $(-1)^{n-k}p_k > 0$ for $k = 0, \ldots, n-1$, where $p_A(\lambda) = \det(A - \lambda \mathrm{I}_n)$ is the characteristics polynomial of A.*

6. *There exists a factorization $A = BB^\top$ with B square and nonsingular (full-rank).*

Schur Complements. A useful technique to check whether a matrix is positive (semi-)definite is to use the Schur Complements. Consider a block-partitioned matrix

$$M = \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix}, \tag{2.1}$$

where A and C are symmetric matrices. If A is invertible, then the Schur complement of A is

$$M/A = C - B^\top A^{-1} B.$$

Similarly, if C is invertible, then the Schur complement of C is

$$M/C = A - BC^{-1}B^\top.$$

We have the following result relating the Schur Complements to positive (semi-)definiteness.

Proposition 2.1 (Schur Complements and PSD)**.** *Consider the block-partitioned matrix M in* (2.1)*,*

- *M is positive definite if and only if both A and M/A are positive definite:*

$$M \succ 0 \Leftrightarrow A \succ 0, M/A = C - B^\top A^{-1} B \succ 0.$$

- *M is positive definite if and only if both C and M/C are positive definite:*

$$M \succ 0 \Leftrightarrow C \succ 0, M/C = A - BC^{-1}B^\top \succ 0.$$

- *If A is positive definite, then M is positive semidefinite if and only if M/A is positive semidefinite:*

$$\text{If } A \succ 0, \text{ then } M \succeq 0 \Leftrightarrow M/A \succeq 0.$$

- *If C is positive definite, then M is positive semidefinite if and only if M/C is positive semidefinite:*

$$\text{If } C \succ 0, \text{ then } M \succeq 0 \Leftrightarrow M/C \succeq 0.$$

2.1.1 Geometric Properties

The set $\mathbb{S}_+^n$ is a proper cone (cf. Definition 1.3). Its interior is $\mathbb{S}_{++}^n$. Under the inner product

$$\langle A, B\rangle = \mathrm{tr}(AB^\top), \quad A, B \in \mathbb{R}^{n\times n},$$

the PSD cone $\mathbb{S}_+^n$ is self-dual.

Next we want to characterize the face of the PSD cone. We first present the following lemma which will turn out to be useful afterwards.

Lemma 2.3 (Range of PSD Matrices)**.** *Let $A, B \in \mathbb{S}_+^n$, then we have*

$$\mathrm{Range}(A) \subseteq \mathrm{Range}(A+B), \tag{2.2}$$

where $\mathrm{Range}(A)$ *denotes the span of the column vectors of A.*

Proof. For any symmetric matrix S, we know

$$\mathrm{Range}(S) = \ker(S)^\perp.$$

Therefore, to prove (2.2), it is equivalent to prove

$$\ker(A) \supseteq \ker(A+B).$$

Pick any $u \in \ker(A+B)$, we have

$$(A+B)u = 0 \Rightarrow u^\top(A+B)u = 0 \Rightarrow u^\top Au + u^\top Bu = 0 \Rightarrow u^\top Au = u^\top Bu = 0,$$

where the last derivation is due to $A, B \succeq 0$. Now that we have $u^\top Au = 0$, we claim that $Au = 0$ must hold, i.e., $u \in \ker(A)$. To see this, write

$$u = \sum_{i=1}^n a_i v_i,$$

where $a_i = \langle u, v_i\rangle$ and $v_i, i = 1, \dots, n$ are the eigenvectors of A corresponding to eigenvalues $\lambda_i, i = 1, \dots, n$. Then we have

$$Au = \sum_{i=1}^n a_i Av_i = \sum_{i=1}^n a_i \lambda_i v_i,$$

and

$$u^\top Au = \sum_{i=1}^n \lambda_i a_i^2 = 0.$$

Since $\lambda_i \geq 0, a_i^2 \geq 0$, we have

$$\lambda_i a_i^2 = 0, \forall i = 1, \dots, n.$$

This indicates that if $\lambda_i > 0$, then $a_i = 0$. Therefore, a_i can only be nonzero for $\lambda_i = 0$, which leads to

$$Au = \sum_{i=1}^n a_i \lambda_i v_i = 0.$$

Therefore, $u \in \ker(A)$, proving the result. □

Lemma 2.3 indicates that if $A \succeq B$, then $\text{Range}(B) \subseteq \text{Range}(A)$. What about the reverse?

Lemma 2.4 (Extend Line Segment)**.** *Let $A, B \in \mathbb{S}_+^n$, if $\text{Range}(B) \subseteq \text{Range}(A)$, then there must exist $C \in \mathbb{S}_+^n$ such that*

$$A \in (B, C),$$

i.e., the line segment from B to A can be extended past A within $\mathbb{S}_+^n$.

Proof. Since $\text{Range}(B) \subseteq \text{Range}(A)$, we have

$$\ker(A) \subseteq \ker(B).$$

Now consider extending the line segment past A to

$$C_\alpha = A + \alpha(A - B) = (1 + \alpha)A - \alpha B,$$

with some $\alpha > 0$. We want to show that there exists $\alpha > 0$ such that $C_\alpha \succeq 0$.

Pick $u \in \mathbb{R}^n$, then either $u \in \ker(B)$ or $u \notin \ker(B)$. If $u \in \ker(B)$, then

$$u^\top C_\alpha u = (1 + \alpha)u^\top Au - \alpha u^\top Bu = (1 + \alpha)u^\top Au \geq 0.$$

If $u \notin \ker(B)$, then due to $\ker(A) \subseteq \ker(B)$, we have $u \notin \ker(A)$ as well. As a result, we have

$$u^\top C_\alpha u = (1 + \alpha)u^\top Au - \alpha u^\top Bu = (1 + \alpha)u^\top Au(1 - \frac{\alpha}{1 + \alpha}\frac{u^\top Bu}{u^\top Au}. \tag{2.3}$$

Since

$$\max_{u:u\notin\ker(A)} \frac{u^\top Bu}{u^\top Au} \leq \frac{\lambda_{\max}(B)}{\lambda_{\min,>0}(A)},$$

where $\lambda_{\min,>0}(A)$ denotes the minimum positive eigenvalue of A, we can always choose α sufficiently small to make (2.3) nonnegative. Therefore, there exists $\alpha > 0$ such that $C_\alpha \succeq 0$. □

In fact, from Lemma 2.3 we can induce a corollary.

Corollary 2.1 (Range of PSD Matrices)**.** *Let $A, B \in \mathbb{S}_+^n$, then we have*

$$\text{Range}(A + B) = \text{Range}(A) + \text{Range}(B),$$

with "+" the Minkowski sum.

Exercise 2.1. Let $A, B \in \mathbb{S}_+^n$, show that $\langle A, B\rangle = 0$ if and only if $\text{Range}(A) \perp \text{Range}(B)$.

For a subset $T \subseteq \mathbb{S}^n_+$, we use $\text{face}(T, \mathbb{S}^n_+)$ to denote the smallest face of $\mathbb{S}^n_+$ that contains T. We first characterize the smallest face that contains a given PSD matrix, i.e., $\text{face}(A, \mathbb{S}^n_+)$ for $A \succeq 0$. Clearly, if A is PD, then $\text{face}(A, \mathbb{S}^n_+) = \mathbb{S}^n_+$ is the entire cone. If A is PSD but singular with rank $r < n$, then A has the following spectral decomposition

$$Q^\top A Q = \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix},$$

where $\Lambda \in \mathbb{S}^r_{++}$ is a diagonal matrix with the r nonzero eigenvalues of A, and $Q \in \mathrm{O}(n)$ is orthogonal. If

$$A = \lambda B + (1-\lambda)C, \quad B, C \in \mathbb{S}^n_+, \lambda \in (0,1),$$

then multiplying both sides by $Q^\top$ and Q we have

$$\begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} = Q^\top A Q = \lambda Q^\top B Q + (1-\lambda) Q^\top C Q.$$

Therefore, it must hold that

$$Q^\top B Q = \begin{bmatrix} B_1 & 0 \\ 0 & 0 \end{bmatrix}, \quad Q^\top C Q = \begin{bmatrix} C_1 & 0 \\ 0 & 0 \end{bmatrix}, \quad B_1 \in \mathbb{S}^r_+, C_1 \in \mathbb{S}^r_+,$$

which is equivalent to

$$B = Q \begin{bmatrix} B_1 & 0 \\ 0 & 0 \end{bmatrix} Q^\top, C = Q \begin{bmatrix} C_1 & 0 \\ 0 & 0 \end{bmatrix} Q^\top, \quad B_1 \in \mathbb{S}^r_+, C_1 \in \mathbb{S}^r_+.$$

We conclude that $\text{face}(A, \mathbb{S}^n_+)$ must contain the set

$$G := \left\{ Q \begin{bmatrix} X & 0 \\ 0 & 0 \end{bmatrix} Q^\top \;\middle|\; X \in \mathbb{S}^r_+ \right\}. \tag{2.4}$$

Exercise 2.2. Show that G in (2.4) is a face of $\mathbb{S}^n_+$, i.e., (i) G is convex; (ii) $u \in (x, y), u \in G, x, y \in \mathbb{S}^n_+ \Rightarrow x, y \in G$.

As a result, we have $\text{face}(A, \mathbb{S}^n_+) = G$.

More general faces of the PSD cone $\mathbb{S}^n_+$ can be characterized as follows (Theorem 3.7.1 in (Wolkowicz et al., 2000)).

Theorem 2.1 (Faces of the PSD Cone)**.** *A set $F \subseteq \mathbb{S}^n_+$ is a face if and only if there exists a subspace $L \subseteq \mathbb{R}^n$ such that*

$$F = \{X \in \mathbb{S}^n_+ \mid \text{Range}(X) \subseteq L\}.$$

Proof. It is easy to prove the "**If**" direction using Lemma 2.3.

First we show F is convex. Pick $A, B \in F$. We have $\text{Range}(A) \subseteq L$ and $\text{Range}(B) \subseteq L$. Let $v_1, \ldots, v_m$ be a set of basis spanning L. We have that, for any $u \in \mathbb{R}^n$,

$$\begin{aligned} Au \in L \Rightarrow Au = \sum_{i=1}^{m} a_i v_i, \\ Bu \in L \Rightarrow Bu = \sum_{i=1}^{m} b_i v_i. \end{aligned} \tag{2.5}$$

So for any $\lambda \in [0, 1]$, we have

$$(\lambda A + (1-\lambda)B)u = \lambda Au + (1-\lambda)Bu = \sum_{i=1}^{m} (\lambda a_i + (1-\lambda) b_i) v_i \in L,$$

implying $\lambda A + (1-\lambda)B \in F$ for any $\lambda \in [0, 1]$.

Now we show that:

$$X \in (A, B), X \in F, A, B \in \mathbb{S}_+^n \Rightarrow A, B \in F.$$

From $X = \lambda A + (1-\lambda)B$ for some $\lambda \in (0, 1)$, and invoking Lemma 2.3, we have

$$\begin{aligned} &\text{Range}(X) = \text{Range}(\lambda A + (1-\lambda)B) \supseteq \text{Range}(\lambda A) = \text{Range}(A) \\ &\text{Range}(X) = \text{Range}(\lambda A + (1-\lambda)B) \supseteq \text{Range}((1-\lambda)B) = \text{Range}(B). \end{aligned} \tag{2.6}$$

Since $\text{Range}(X) \subseteq L$ due to $X \in F$, we have

$$\text{Range}(A) \subseteq L, \quad \text{Range}(B) \subseteq L,$$

leading to $A, B \in F$.

The proof for the "**Only If**" direction can be found in Theorem 3.7.1 of (Wolkowicz et al., 2000). □

2.2 Semidefinite Programming

2.2.1 Spectrahedra

Recall the definition of a polyhedron in (1.14), i.e., a vector x constrained by finitely many linear inequalities. The feasible set of a Linear Program is a polyhedron.

Similarly, we define a **spectrahedron** as a set defined by finitely many **linear matrix inequalities** (LMIs). Spectrahedra are the feasible sets of Semidefinite Programs (SDPs).

A linear matrix inequality has the form

$$A_0 + \sum_{i=1}^{m} A_i x_i \succeq 0,$$

where $A_i \in \mathbb{S}^n, i = 0, \dots, m$ are given symmetric matrices. Correspondingly, a spectrahedron is defined by finitely many LMIs.

Definition 2.1 (Spectrahedron)**.** A set $S \subseteq \mathbb{R}^m$ is a spectrahedron if it has the form

$$S = \left\{ x \in \mathbb{R}^m \;\middle|\; A_0 + \sum_{i=1}^{m} x_i A_i \succeq 0 \right\},$$

for given symmetric matrices $A_0, A_1, \dots, A_m \in \mathbb{S}^n$.

Note that there is no less of generality in defining a spectrahedron using a single LMI. For example, in the case of a set defined by two LMIs:

$$S = \left\{ x \in \mathbb{R}^m \;\middle|\; A_0 + \sum_{i=1}^{m} x_i A_i \succeq 0, B_0 + \sum_{i=1}^{m} x_i B_i \succeq 0 \right\}, A_i \in \mathbb{S}^n, B_i \in \mathbb{S}^d,$$

we can compress the two LMIs into a single LMI by putting A_i and B_i along the diagonal:

$$S = \left\{ x \in \mathbb{R}^m \;\middle|\; \begin{bmatrix} A_0 & \\ & B_0 \end{bmatrix} + \sum_{i=1}^{m} x_i \begin{bmatrix} A_i & \\ & B_i \end{bmatrix} \succeq 0 \right\}.$$

Leveraging (5) of Lemma 2.1, we know that a PSD constraint is equivalent to weakly alternating signs of the characteristic polynomial of the given matrix. Therefore, a spectrahedron is defined by finitely many polynomial inequalities, i.e., a spectrahedron is a (convex) **basic semialgebraic set**, as seen in the following example (Blekherman et al., 2012).

Example 2.1 (Elliptic Curve)**.** Consider the spectrahedron in $\mathbb{R}^2$ defined by

$$\left\{ (x, y) \in \mathbb{R}^2 \;\middle|\; A(x, y) = \begin{bmatrix} x+1 & 0 & y \\ 0 & 2 & -x-1 \\ y & -x-1 & 2 \end{bmatrix} \succeq 0 \right\}.$$

To obtain scalar inequalities defining the set, let

$$p_A(\lambda) = \det(\lambda I - A(x, y)) = \lambda^3 + p_2\lambda^2 + p_1\lambda + p_0$$

be the characteristic polynomial of $A(x, y)$. $A(x, y) \succeq 0$ is then equivalent to the coefficients weakly alternating in sign:

$$\begin{aligned} p_2 &= -(x + 5) \leq 0, \\ p_1 &= -x^2 + 2x - y^2 + 7 \geq 0, \\ p_0 &= -(3 + x - x^3 - 3x^2 - 2y^2) \leq 0. \end{aligned} \tag{2.7}$$

We can use the following Matlab script to plot the set shown in Fig. 2.2. (The code is also available at here.) As we can see, the spectrahedron is convex, but it is not a polyhedron.

```
x = -2:0.01:2;
y = -2:0.01:2;
[X,Y] = meshgrid(x,y);

ineq = (-X - 5 <= 0) & ...
    (-X.^2 + 2*X - Y.^2 + 7 >=0) & ...
    (3 + X - X.^3 - 3*X.^2 - 2*Y.^2 >= 0);

h = pcolor(X,Y,double(ineq)) ;
h.EdgeColor = 'none' ;
```

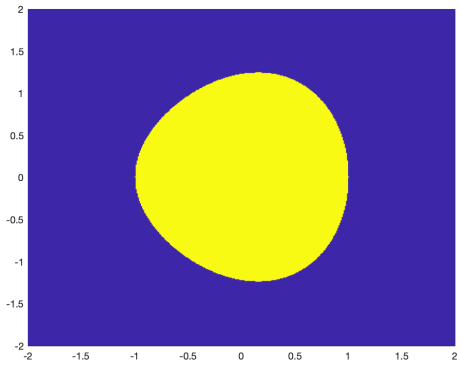

Figure 2.2: Elliptic Curve.

We can use the same technique to visualize the elliptope, a spectrahedron that we will see again later when we study the MAXCUT problem.

Example 2.2 (Elliptope)**.** Consider the 3D elliptope defined by

$$\left\{ (x,y,z) \in \mathbb{R}^3 \;\middle|\; A(x,y,z) = \begin{bmatrix} 1 & x & y \\ x & 1 & z \\ y & z & 1 \end{bmatrix} \succeq 0 \right\}.$$

The characteristic polynomial of $A(x,y,z)$ is

$$p_A(\lambda) = \lambda^3 - 3\lambda^2 + (-x^2 - y^2 - z^2 + 3)\lambda + x^2 - 2xyz + y^2 + z^2 - 1.$$

The coefficients need to weakly alternative in sign, we have the inequalities

$$\begin{aligned} -x^2 - y^2 - z^2 + 3 &\geq 0 \\ x^2 - 2xyz + y^2 + z^2 - 1 &\leq 0 \end{aligned} \tag{2.8}$$

Using the Matlab script here, we generate the following plot.

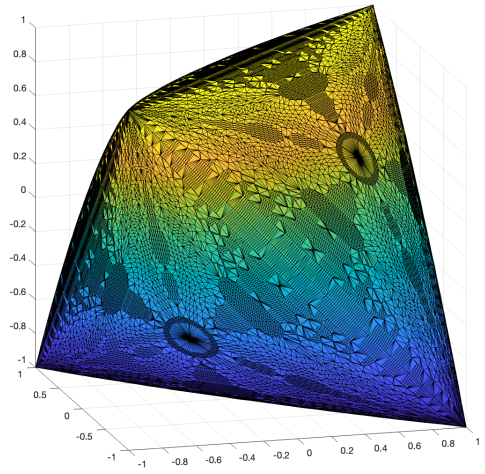

Figure 2.3: Elliptope.

Another example is provided in Fig. 2.1.

2.2.2 Formulation and Duality

Semidefinite programs (SDPs) are linear optimization problems over spectrahedra. A standard SDP in **primal** form is written as

$$\boxed{\begin{aligned} p^\star = \min_{X \in \mathbb{S}^n} \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \mathcal{A}(X) = b, \\ & X \succeq 0 \end{aligned}} \tag{2.9}$$

where $C \in \mathbb{S}^n$, $b \in \mathbb{R}^m$, and the linear map $\mathcal{A} : \mathbb{S}^n \to \mathbb{R}^m$ is defined as

$$\mathcal{A}(X) := \begin{bmatrix} \langle A_1, X \rangle \\ \vdots \\ \langle A_i, X \rangle \\ \langle A_m, X \rangle \end{bmatrix}.$$

Recall that $\langle C, X\rangle = \mathrm{tr}(CX)$. The feasible set of (2.9) is the intersection of the PSD cone ($\mathbb{S}^n_+$) and the affine subspace defined by $\mathcal{A}(X) = b$.

Closely related to the primal SDP (2.9) is the **dual** problem

$$\boxed{\begin{array}{lll} d^\star = \max\limits_{y\in\mathbb{R}^m} & \langle b, y\rangle \\ \text{s.t.} & C - \mathcal{A}^*(y) \succeq 0 \end{array}} \tag{2.10}$$

where $\mathcal{A}^* : \mathbb{R}^m \to \mathbb{S}^n$ is the **adjoint** map defined as

$$\mathcal{A}^*(y) := \sum_{i=1}^m y_i A_i.$$

Observe how the primal-dual SDP pair (2.9)-(2.10) parallels the primal-dual LP pair (1.17)-(1.18).

Weak duality. We have a similar weak duality between the primal and dual. Pick any X that is feasible for the primal (2.9) and y that is feasible for the dual (2.10), we have

$$\boxed{\langle C, X\rangle - \langle b, y\rangle = \langle C, X\rangle - \langle \mathcal{A}(X), y\rangle = \langle C - \mathcal{A}^*(y), X\rangle \geq 0,}$$

where the last inequality holds because both $C - \mathcal{A}^*(y)$ and X are positive semidefinite. As a result, we have the weak duality

$$d^\star \leq p^\star.$$

Similar to the LP case, we will denote $p^\star = +\infty$ if the primal is infeasible, $p^\star = -\infty$ if the primal is unbounded below. We will denote $d^\star = +\infty$ if the dual is unbounded above, and $d^\star = -\infty$ if the dual is infeasible. We say the primal (or the dual) is **solvable** if it admits optimizers. We denote $p^\star - d^\star$ as the **duality gap**.

Recall Theorem 1.9 states that in LP, if at least one of the primal and dual is feasible, then strong duality holds (i.e., $p^\star = d^\star = \{\pm\infty, \text{finite}\}$). Unfortunately, this does not carry over to SDPs. Let us provide several examples.

Example 2.3 (Failure of SDP Strong Duality)**.** The first example, from (Ramana, 1997), shows that even if both primal and dual are feasible, there could exist a nonzero duality gap. Consider the following SDP pair for some $\alpha \geq 0$

$$\begin{cases} \min_{X\in\mathbb{S}^3} & \alpha X_{11} \\ \text{s.t.} & X_{22} = 0 \\ & X_{11} + 2X_{23} = 1 \\ & \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ * & X_{22} & X_{23} \\ * & * & X_{33} \end{bmatrix} \succeq 0 \end{cases}, \quad \begin{cases} \max_{y\in\mathbb{R}^2} & y_2 \\ \text{s.t.} & \begin{bmatrix} \alpha & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \succeq \begin{bmatrix} y_2 & 0 & 0 \\ 0 & y_1 & y_2 \\ 0 & y_2 & 0 \end{bmatrix} \end{cases}$$

To examine the primal feasible set, let us pick the bottom-right 2×2 submatrix of X. The determinant of this submatrix needs to be nonnegative (due to (4) of Lemma 2.1):

$$X_{22}X_{33} - X_{23}^2 \geq 0.$$

Because $X_{22} = 0$, we have $X_{23} = 0$ and hence $X_{11} = 1$. Therefore, $p^\star = \alpha$ is attained.

To examine the dual feasible set, pick the bottom-right 2×2 submatrix of

$$\begin{bmatrix} \alpha - y_2 & 0 & 0 \\ 0 & -y_1 & -y_2 \\ 0 & -y_2 & 0 \end{bmatrix} \succeq 0,$$

we have $y_2 = 0$. As a result, $d^\star = 0$, and strong duality fails.

The second example, from (Todd, 2001), shows that the duality gap can even be infinite. Consider the primal-dual SDP

$$\begin{cases} \min_{X \in \mathbb{S}^2} & 0 \\ \text{s.t.} & X_{11} = 0 \\ & X_{12} = 1 \\ & \begin{bmatrix} X_{11} & X_{12} \\ * & X_{22} \end{bmatrix} \succeq 0 \end{cases}, \begin{cases} \max_{y \in \mathbb{R}^2} & 2y_2 \\ \text{s.t.} & \begin{bmatrix} -y_1 & -y_2 \\ -y_2 & 0 \end{bmatrix} \succeq 0 \end{cases}$$

Clearly, the primal is infeasible because

$$\begin{bmatrix} 0 & 1 \\ 1 & X_{22} \end{bmatrix}$$

can never be PSD. So $p^\star = +\infty$. The dual problem, however, is feasible. From the PSD constraint we have $y_2 = 0$ and $d^\star = 0$. Therefore, the duality gap is infinite.

The third example, from (Todd, 2001), shows that even when the duality gap is zero, the primal or dual problem may not admit optimizers. Consider the primal-dual SDP

$$\begin{cases} \min_{X \in \mathbb{S}^2} & 2X_{12} \\ \text{s.t.} & -X_{11} = -1 \\ & -X_{22} = 0 \\ & \begin{bmatrix} X_{11} & X_{12} \\ * & X_{22} \end{bmatrix} \succeq 0 \end{cases}, \begin{cases} \max_{y \in \mathbb{R}^2} & -y_1 \\ \text{s.t.} & \begin{bmatrix} y_1 & 1 \\ 1 & y_2 \end{bmatrix} \succeq 0 \end{cases}$$

To examine the primal feasible set, we have

$$\begin{bmatrix} 1 & X_{12} \\ X_{12} & 0 \end{bmatrix} \succeq 0$$

implies $X_{12} = 0$. Hence the primal feasible set only has one point and $p^\star = 0$. The dual feasible set reads

$$y_1 y_2 \geq 1, \quad y_1 \geq 0, \quad y_2 \geq 0,$$

and we want to minimize y_1. Clearly, $d^\star = 0$ but it is not attainable. Therefore, strong duality holds but the dual problem is not solvable.

A Matlab script that passes these three examples to SDP solvers can be found here.

The examples above are somewhat "pathological" and they show that SDPs in general can be more complicated that LPs. It turns out, with the addition of **Slater's condition**, i.e., **strict feasibility** of the primal and dual, we can recover nice results parallel to those of LP.

Theorem 2.2 (SDP Strong Duality)**.** *Assume both the primal SDP* (2.9) *and the dual SDP* (2.10) *are* strictly feasible*, i.e., there exists $X \succ 0$ such that $\mathcal{A}(X) = b$ for the primal and there exists $y \in \mathbb{R}^m$ such that $C - \mathcal{A}^*(y) \succ 0$ for the dual, then strong duality holds, i.e., both problems are solvable and admit optimizers, and $p^\star = d^\star$ equals to some finite number.*

Further, a pair of primal-dual feasible points (X, y) is optimal if and only if

$$\langle C, X\rangle = \langle b, y\rangle \Leftrightarrow \langle C - \mathcal{A}^*(y), X\rangle = 0 \Leftrightarrow (C - \mathcal{A}^*(y))X = 0.$$

One can relax the requirement of both primal and dual being strictly feasible to only one of them being strictly feasible, and similar results would hold. Precisely, if the primal is bounded below and strictly feasible, then $p^\star = d^\star$ and the dual is solvable. If the dual is bounded above and strictly feasible, then $p^\star = d^\star$ and the primal is solvable (Nie, 2023).

Example 2.4 (SDP Strong Duality)**.** Consider the following primal-dual SDP pair

$$\begin{cases} \min_{X\in\mathbb{S}^2} & 2X_{11} + 2X_{12} \\ \text{s.t.} & X_{11} + X_{22} = 1 \\ & \begin{bmatrix} X_{11} & X_{12} \\ * & X_{22} \end{bmatrix} \succeq 0 \end{cases}, \quad \begin{cases} \max_{y\in\mathbb{R}} & y \\ \text{s.t.} & \begin{bmatrix} 2-y & 1 \\ 1 & -y \end{bmatrix} \succeq 0 \end{cases}$$

Choose

$$X = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \succ 0$$

we see the primal is strictly feasible. Choose $y = -1$, we have

$$\begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix} \succ 0$$

and the dual is strictly feasible. Therefore, strong duality holds.

In this case, pick the pair of primal-dual feasible points

$$X^\star = \begin{bmatrix} \frac{2-\sqrt{2}}{4} & -\frac{1}{2\sqrt{2}} \\ -\frac{1}{2\sqrt{2}} & \frac{2+\sqrt{2}}{4} \end{bmatrix}, \quad y^\star = 1 - \sqrt{2},$$

we have

$$\langle C, X^\star \rangle = 1 - \sqrt{2} = \langle b, y^\star \rangle,$$

and both $X^\star$ and $y^\star$ are optimal.

2.2.3 Geometric Properties

2.3 Software for Conic Optimization

Linear optimization over the nonnegative orthant ($\mathbb{R}^n_+$), the second-order cone ($\mathcal{Q}_n$), and the positive semidefinite cone ($\mathbb{S}^n_+$) forms the foundation of modern convex optimization, commonly referred to as conic optimization. These three types of cones are self-dual, and there exist efficient algorithms to solve the convex optimization problems. Popular solvers include SDPT3, SeDuMi, MOSEK, and SDPNAL+.

In this section, we introduce how we should "talk to" the numerical solvers for conic optimization, i.e., how should we pass a mathematically written conic optimization to a numerical solver. Note that in many cases, this "transcription" can be done by programming packages such as CVX, CVXPY, YALMIP etc., but I think it is important to understand the standard interface of numerical solvers because (i) it reinforces our understanding of the mathematical basics, (ii) it gets us closer to designing custom numerical solvers for specific problems, (iii) if you are a heavy convex optimization user you will realize that many of the programming packages are not "efficient" in transcribing the original optimization problem (but they are indeed very general). I have had cases where solving the conic optimization takes a few minutes but transcribing the problem to the solver takes half an hour.

We will use the SeDuMi format as an example. Consider the following general linear convex optimization problem

$$\begin{aligned} \max_{y \in \mathbb{R}^m} \quad & b^\top y \\ \text{s.t.} \quad & Fy = g \\ & f \geq Gy \\ & h_i^\top y + \tau_i \geq \|H_i y + p_i\|_2, i = 1, \ldots, r \\ & B_{j,0} + \sum_{k=1}^m y_k B_{j,k} \succeq 0, j = 1, \ldots, s, \end{aligned} \tag{2.11}$$

for given matrices and vectors

$F \in \mathbb{R}^{\ell_1 \times m}, G \in \mathbb{R}^{\ell_2 \times m}, H_i \in \mathbb{R}^{l_i \times m}, B_{j,k} \in \mathbb{S}^{n_j}, h_i \in \mathbb{R}^m, p_i \in \mathbb{R}^{l_i}, \tau_i \in \mathbb{R}, g \in \mathbb{R}^{\ell_1}, f \in \mathbb{R}^{\ell_2}$.

Define the linear function

$$\phi(y) := \left(Fy, Gy, \begin{bmatrix} -h_1^\top y \\ -H_1 y \end{bmatrix}, \dots, \begin{bmatrix} -h_r^\top y \\ -H_r y \end{bmatrix}, -\sum_{k=1}^m y_k B_{1,k}, \dots, -\sum_{k=1}^m y_k B_{s,k} \right),$$

which is a linear map from $\mathbb{R}^m$ to the vector space of Cartesian products

$$V := \mathbb{R}^{\ell_1} \times \mathbb{R}^{\ell_2} \times \mathbb{R}^{l_1+1} \times \cdots \times \mathbb{R}^{l_r+1} \times \mathbb{S}^{n_1} \times \cdots \times \mathbb{S}^{n_s}.$$

A vector $X \in V$ can be written as a tuple

$$X = (x_1, x_2, \mathrm{x}_1, \dots, \mathrm{x}_r, X_1, \dots, X_s).$$

Given another vector $Y \in V$

$$X = (y_1, y_2, \mathrm{y}_1, \dots, \mathrm{y}_r, Y_1, \dots, Y_s),$$

the inner product between X and Y is defined as

$$\langle X, Y \rangle = \langle x_1, y_2 \rangle + \langle x_2, y_2 \rangle + \sum_{i=1}^r \langle \mathrm{x}_i, \mathrm{y}_i \rangle + \sum_{j=1}^s \langle X_j, Y_j \rangle.$$

Let $\mathcal{K}$ be the Cartesian product of the free cone, the nonnegative orthant, the second-order cone, and the PSD cone

$$\mathcal{K} := \mathbb{R}^{\ell_1} \times \mathbb{R}_+^{\ell_2} \times \mathcal{Q}_{l_1} \times \cdots \times \mathcal{Q}_{l_r} \times \mathbb{S}_+^{n_1} \times \cdots \times \mathbb{S}_+^{n_s}.$$

Its dual cone is

$$\mathcal{K}^* = \{0\}^{\ell_1} \times \times \mathbb{R}_+^{\ell_2} \times \mathcal{Q}_{l_1} \times \cdots \times \mathcal{Q}_{l_r} \times \mathbb{S}_+^{n_1} \times \cdots \times \mathbb{S}_+^{n_s}.$$

Note that all the cones there are self-dual except the free cone whose dual is the zero point. Then denote

$$C := \left(g, f, \begin{bmatrix} \tau_1 \\ p_1 \end{bmatrix}, \dots, \begin{bmatrix} \tau_r \\ p_r \end{bmatrix}, B_{1,0}, \dots, B_{s,0} \right) \in V,$$

we have that the original optimization (2.11) is simply the following dual conic problem

$$\max_{y \in \mathbb{R}^m} \{ b^\top y \mid C - \phi(y) \in \mathcal{K}^* \}. \tag{2.12}$$

The linear map $\phi(y)$ can be written as

$$\phi(y) = y_1 A_1 + \dots, y_m A_m$$

for vectors $A_1, \dots, A_m$ in the space V. Therefore, the primal problem to (2.12) is

$$\min_{X \in V} \{ \langle C, X \rangle \mid \langle A_i, X \rangle = b_i, i = 1, \dots, m, X \in \mathcal{K} \}. \tag{2.13}$$

Let us practice an example from (Nie, 2023).

Example 2.5 (SeDuMi Example)**.** Consider the following optimization problem as an instance of (2.11):

$$\begin{array}{ll} \max\limits_{y\in\mathbb{R}^3} & y_3 - y_1 \\ \text{s.t.} & y_1 + y_2 + y_3 = 3 \\ & -1 \geq -y_1 - y_2 \\ & -1 \geq -y_2 - y_3 \\ & y_1 + y_3 \geq \sqrt{(y_1-1)^2 + y_2^2 + (y_3-1)^2} \\ & \begin{bmatrix} 1 & y_1 & y_2 \\ y_1 & 2 & y_3 \\ y_2 & y_3 & 3 \end{bmatrix} \succeq 0 \end{array} \tag{2.14}$$

Clearly we have $b = (-1, 0, 1)$. The linear map $\phi(y)$ is

$$\phi(y) = \left(y_1 + y_2 + y_3, \begin{bmatrix} -y_1 - y_2 \\ -y_2 - y_3 \end{bmatrix}, \begin{bmatrix} -y_1 - y_3 \\ -y_1 \\ -y_2 \\ -y_3 \end{bmatrix}, \begin{bmatrix} 0 & -y_1 & -y_2 \\ -y_1 & 0 & -y_3 \\ -y_2 & -y_3 & 0 \end{bmatrix} \right).$$

The vector space $V = \mathbb{R} \times \mathbb{R}^2 \times \mathbb{R}^4 \times \mathbb{S}^3$, and the cone $\mathcal{K}$ is

$$\mathcal{K} = \mathbb{R} \times \mathbb{R}^2_+ \times \mathcal{Q}_3 \times \mathbb{S}^3_+.$$

The vectors A_1, A_2, A_3 are

$$\begin{aligned} A_1 &= \left(1, \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right) \\ A_2 &= \left(1, \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \right) \\ A_3 &= \left(1, \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix} \right). \end{aligned} \tag{2.15}$$

The vector C is

$$C = \left(3, \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \begin{bmatrix} 0 \\ -1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \right).$$

To input this problem to SeDuMi, we only need to provide the data (A, b, C) together with the description of the cones $\mathcal{K}$.

```
% describe dimensions of the cones
K.f = 1; % free cone
K.l = 2; % nonnegative orthant
K.q = 4; % second order cone
K.s = 3; % psd cone

% provide A,b,c
c = [3,-1,-1,0,-1,0,-1,1,0,0,0,2,0,0,0,3];
b = [-1,0,1];
A = [
    1,-1,0,-1,-1,0,0,0,-1,0,-1,0,0,0,0,0;
    1,-1,-1,0,0,-1,0,0,0,-1,0,0,0,-1,0,0;
    1,0,-1,-1,0,0,-1,0,0,0,0,0,-1,0,-1,0
    ];

% solve using sedumi
[xopt,yopt,info] = sedumi(A,b,c,K);
```

Note that in providing A, we vectorize each A_i and place it along the i-th row of the matrix A. The above Matlab script gives us the optimal solution

$$y^\star = (0, 1, 2).$$

To run the code, make sure you download SeDuMi and add that to your Matlab path.

2.4 Interior Point Algorithm

A nice property of semidefinite optimization is that it can be solved in polynomial time. The algorithm of choice for solving SDPs is called an **interior point method** (IPM), which is also what popular SDP solvers like SeDuMI, MOSEK, and SDPT3 implement under the hood.

Although it can be difficult to implement an SDP solver as efficient and robust as MOSEK (as it requires many engineering wisdom), it is surprisingly simple to sketch out the basic algorithmic framework, as it is based on Newton's method for solving a system of nonlinear equations. Before I introduce you the basic algorithm, let me review two useful preliminaries.

Newton's Method

Given a function $f : \mathbb{R} \to \mathbb{R}$ that is continuously differentiable, Newton's method is designed to find a root of $f(x) = 0$. Given an initial iterate $x^{(0)}$, Newton's method works as follows

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})},$$

where $f'(x^{(k)})$ denotes the derivative of f at the current iterate $x^{(k)}$. This simple algorithm is indeed (in my opinion) the most important foundation of modern numerical optimization (Nocedal and Wright, 1999). Under mild conditions, Newton's method has at least quadratic convergence rate, that is to say, if $|x^{(k)} - x^\star| = \epsilon$, then $|x^{(k+1)} - x^\star| = O(\epsilon^2)$. Of course, there exist pathological cases where even linear convergence is not guaranteed (e.g., when $f'(x^\star) = 0$).

Newton's method can be generalized to find a point at which multiple functions vanish simultaneously. Given a function $F : \mathbb{R}^n \to \mathbb{R}^n$ that is continuously differentiable, and an initial iterate $x^{(0)}$, Newton's method reads

$$x^{(k+1)} = x^{(k)} - J_F(x^{(k)})^{-1} F(x^{(k)}), \tag{2.16}$$

where $J_F(\cdot)$ denotes the Jacobian of F. Iteration (2.16) is equivalent to

$$\begin{aligned} J_F(x^{(k)})\Delta x^{(k)} &= -F(x^{(k)}) \\ x^{(k+1)} &= x^{(k)} + \Delta x^{(k)} \end{aligned} \tag{2.17}$$

i.e., one first solves a linear system of equations to find an update direction $\Delta x^{(k)}$, and then take a step along the direction.

As we will see, IPMs for solving SDPs can be interpreted as applying Newton's method to the perturbed KKT optimality conditions.

We introduce another useful preliminary about the symmetric Kronecker product.

Symmetric Vectorization and Kronecker Product

Consider a linear operator on $\mathbb{R}^{n\times n}$:

$$\begin{aligned} \mathbb{R}^{n\times n} &\to \mathbb{R}^{n\times n} \\ K &\mapsto NKM^\top \end{aligned} \tag{2.18}$$

where $N, M \in \mathbb{R}^{n\times n}$ are given square matrices. This linear map is equivalent to

$$\begin{aligned} \mathbb{R}^{n^2} &\to \mathbb{R}^{n^2} \\ \mathrm{vec}(K) &\mapsto (M \otimes N)\mathrm{vec}(K) \end{aligned} \tag{2.19}$$

where $\otimes$ denotes the usual kronecker product.

Symmetric vectorization and kronecker product is to generalize the above linear map on $\mathbb{R}^{n\times n}$ to $\mathbb{S}^n$.

Consider a linear operator on $\mathbb{S}^n$:

$$\begin{aligned} \mathbb{S}^n &\to \mathbb{S}^n \\ K &\mapsto \frac{1}{2}\left(NKM^\top + MKN^\top\right) \end{aligned} \tag{2.20}$$

where $N, M \in \mathbb{R}^{n\times n}$ are given square matrices. This linear map is equivalent to

$$\begin{aligned} \mathbb{R}^{n^\Delta} &\to \mathbb{R}^{n^\Delta} \\ \mathrm{svec}(K) &\mapsto (M \otimes_{\mathrm{s}} N)\mathrm{svec}(K) \end{aligned} \tag{2.21}$$

where

$$n^\Delta = \frac{n(n+1)}{2}$$

is the n-th triangle number, $\mathrm{svec}(K)$ denotes the symmetric vectorization of K defined as

$$\mathrm{svec}(K) = \begin{bmatrix} K_{11} \\ \sqrt{2}K_{12} \\ \vdots \\ \sqrt{2}K_{1n} \\ K_{22} \\ \vdots \\ \sqrt{2}K_{2n} \\ \vdots \\ K_{nn} \end{bmatrix},$$

and $M \otimes_{\mathrm{s}} N$ denotes the symmetric kronecker product. Note that we have

$$\langle A, B\rangle = \langle \mathrm{svec}(A), \mathrm{svec}(B)\rangle, \quad \forall A, B \in \mathbb{S}^n,$$

and

$$M \otimes_{\mathrm{s}} N = N \otimes_{\mathrm{s}} M.$$

To compute the symmetric kronecker product, see (Schacke, 2004). The $\otimes_{\mathrm{s}}$ function is readily implemented in software packages such as SDPT3.

The following property of the symmetric kronecker product is useful for us later.

Lemma 2.5 (Spectrum of Symmetric Kronecker Product)**.** *Let $M, N \in \mathbb{S}^n$ be two symmetric matrices that commute, i.e., $MN = NM$, and $\alpha_1, \dots, \alpha_n$ and $\beta_1, \dots, \beta_n$ be their eigenvalues with $v_1, \dots, v_n$ a common basis of orthonormal eigenvectors. The n^Δ eigenvalues of $M \otimes_{\mathrm{s}} N$ are given by*

$$\frac{1}{2}(\alpha_i\beta_j + \beta_i\alpha_j), \quad 1 \le i \le j \le n,$$

with the corresponding set of orthonormal eigenvectors

$$\begin{cases} \mathrm{svec}(v_i v_i^\top) & \text{if } i = j \\ \frac{1}{\sqrt{2}}\mathrm{svec}(v_i v_j^\top + v_j v_i^\top) & \text{if } i < j \end{cases}.$$

It is easy to see that, if M, N are two PSD matrices that commute, then $M \otimes_{\mathrm{s}} N$ is also PSD.

For more properties of symmetric vectorization and Kronecker product, see (Schacke, 2004).

Now we are ready to sketch the interior-point algorithm.

2.4.1 The Central Path

Assuming strict feasibility, strong duality holds between the SDP primal (2.9) and dual (2.10), then (X, y, Z) is primal-dual optimal if and only if they satisfy the following KKT optimality conditions

$$\begin{aligned} \mathcal{A}(X) = b, \quad X \succeq 0 \\ C - \mathcal{A}^*(y) - Z = 0, \quad Z \succeq 0 \\ \langle X, Z \rangle = 0 \Leftrightarrow XZ = 0 \end{aligned} \tag{2.22}$$

The first idea in IPM is to relax the last condition in (2.22), i.e., zero duality gap, to a small positive gap, which leads to the notion of a central path.

Definition 2.2 (Central Path)**.** A point (X^μ, y^μ, Z^μ) is said to lie on the central path if there exists $\mu > 0$ such that

$$\begin{aligned} \mathcal{A}(X^\mu) = b, \quad X^\mu \succeq 0 \\ C - \mathcal{A}^*(y^\mu) - Z^\mu = 0, \quad Z^\mu \succeq 0, \\ X^\mu Z^\mu = \mu \mathrm{I} \end{aligned} \tag{2.23}$$

that is, (X^μ, y^μ, Z^μ) is primal and dual feasible, but attain a nonzero duality gap:

$$\langle C, X^\mu \rangle - \langle b, y^\mu \rangle = \langle C, X^\mu \rangle - \langle \mathcal{A}(X^\mu), y^\mu \rangle = \langle C - \mathcal{A}^*(y^\mu), X^\mu \rangle = \langle X^\mu, Z^\mu \rangle = n\mu.$$

The central path exists and is unique.

Theorem 2.3 (Central Path)**.** *Assume the SDP pair* (2.9) *and* (2.10) *are strictly feasible. For any* $\mu > 0$*,* (X^μ, y^μ, Z^μ) *satisfying* (2.23) *exists and is unique. Moreover,*

$$(X, y, Z) = \lim_{\mu \to 0} (X^\mu, y^\mu, Z^\mu)$$

exists and solves (2.9) *and* (2.10)*.*

Theorem 2.3 states that the limit of the central path leads to a solution of the SDP. Therefore, the basic concept of IPM is to follow the central path and converge to the optimal solution.

2.4.2 The AHO Newton Direction

We will focus on the central path equation (2.23) and for simplicity of notation, we will rename (X^μ, y^μ, Z^μ) as (X, y, Z). Discarding the positive semidefiniteness condition for now, we can view (2.23) as finding a root to the following function

$$F(X, y, Z) = \begin{pmatrix} \mathcal{A}^*(y) + Z - C \\ \mathcal{A}(X) - b \\ XZ - \mu \mathrm{I} \end{pmatrix} = 0. \tag{2.24}$$

However, there is an issue with F defined as above. The input of the function (X, y, Z) lives in $\mathbb{S}^n \times \mathbb{R}^m \times \mathbb{S}^n$, but the output lives in $\mathbb{S}^n \times \mathbb{R}^m \times \mathbb{R}^{n\times n}$ because XZ is not guaranteed to be symmetric (two symmetric matrices may not commute). Therefore, Newton's method cannot be directly applied.

An easy way to fix this issue is to treat the input and output of F as both $\mathbb{R}^{n\times n} \times \mathbb{R}^m \times \mathbb{R}^{n\times n}$. This could work, however, leads to nonsymmetric Newton iterates.

(Alizadeh et al., 1998) proposed a better idea – to equivalently rewrite (2.24) as

$$F(X, y, Z) = \begin{pmatrix} \mathcal{A}^*(y) + Z - C \\ \mathcal{A}(X) - b \\ \frac{1}{2}(XZ + ZX) - \mu \mathrm{I} \end{pmatrix} = 0. \tag{2.25}$$

The next proposition states that (2.25) and (2.24) are indeed equivalent.

Proposition 2.2 (Symmetrized Central Path)**.** *If $X, Z \succeq 0$, then*

$$XZ = \mu \mathrm{I} \Leftrightarrow XZ + ZX = 2\mu \mathrm{I}.$$

Proof. The $\Rightarrow$ direction is obvious. To show the " " direction, write $X = Q\Lambda Q^\top$ with $QQ^\top = \mathrm{I}$. □

Now that the input and output domains of F in (2.25) are both $\mathbb{S}^n \times \mathbb{R}^m \times \mathbb{S}^n$, we can apply Newton's method to solving the system of equations. For ease of implementation, let us denote

$$x = \mathrm{svec}(X), \quad z = \mathrm{svec}(Z), \quad c = \mathrm{svec}(C),$$

$$A^\top = \begin{bmatrix} \mathrm{svec}(A_1) & \mathrm{svec}(A_2) & \cdots & \mathrm{svec}(A_m) \end{bmatrix}$$

and rewrite (2.25) as

$$F(x, y, z) = \begin{pmatrix} A^\top y + z - c \\ Ax - b \\ \frac{1}{2}\mathrm{svec}(XZ + ZX) - \mathrm{svec}(\mu \mathrm{I}) \end{pmatrix} = 0. \tag{2.26}$$

The Jacobian of F reads

$$J_F = \begin{bmatrix} 0 & A^\top & \mathrm{I} \\ A & 0 & 0 \\ P & 0 & Q \end{bmatrix} \tag{2.27}$$

where

$$P = Z \otimes_{\mathrm{s}} \mathrm{I}, \quad Q = X \otimes_{\mathrm{s}} \mathrm{I},$$

due to the symmetric kronecker product introduced before

$$\frac{1}{2}\mathrm{svec}(XZ + ZX) = (Z \otimes_{\mathrm{s}} \mathrm{I})x = (X \otimes_{\mathrm{s}} \mathrm{I})z.$$

Let

$$r_d = c - A^\top y - z, \quad r_p = b - Ax, \quad r_c = \operatorname{svec}(\mu \mathrm{I}) - \frac{1}{2}\operatorname{svec}(XZ + ZX),$$

be the dual, primal, and complementarity residuals, applying Newton's method to (2.26) gives us the Newton direction as the solution to the following linear system

$$\begin{bmatrix} 0 & A^\top & \mathrm{I} \\ A & 0 & 0 \\ P & 0 & Q \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} r_d \\ r_p \\ r_c \end{bmatrix}. \tag{2.28}$$

One can directly form the block matrix in (2.28) and solve the linear system. However, leveraging the sparsity of the linear system, we can do better.

We can first use the third equation to eliminate Δz:

$$\Delta z = Q^{-1}(r_c - P\Delta x). \tag{2.29}$$

Then we can use the first equation to eliminate Δx:

$$\Delta x = -P^{-1}Qr_d + P^{-1}r_c + P^{-1}QA^\top \Delta y. \tag{2.30}$$

Then we are left with a single equation of Δy:

$$\underbrace{AP^{-1}QA^\top}_{M} \Delta y = r_p + AP^{-1}(Qr_d - r_c), \tag{2.31}$$

which is called the **Schur system**.

2.4.3 Basic Algorithm

With the AHO Newton direction worked out above, we can formulate the basic primal-dual path following interior point algorithm.

1. Choose X, Z strictly feasible for (2.9) and (2.10), $0 \le \sigma < 1$, define
$$\mu = \sigma \frac{\langle X, Z \rangle}{n}$$
2. Solve the Newton direction from (2.31), (2.30), and (2.29)
3. Step along the Newton direction
$$x \leftarrow x + \alpha \Delta x, \quad y \leftarrow y + \beta \Delta y, \quad z \leftarrow z + \beta \Delta z,$$
where the step sizes α, β are chosen as
$$\alpha = \min(1, \tau\hat{\alpha}), \quad \beta = \min(1, \tau\hat{\beta})$$
with $\tau \in (0, 1)$ and $\hat{\alpha}, \hat{\beta}$ computed by
$$\hat{\alpha} = \sup\{\bar{\alpha} \mid x + \bar{\alpha}\Delta x \succeq 0\}, \quad \hat{\beta} = \sup\{\bar{\beta} \mid z + \bar{\beta}\Delta z \succeq 0\}. \tag{2.32}$$

To compute the maximum step sizes $\hat{\alpha}$ and $\hat{\beta}$ in (2.32), let

$$X = \text{smat}(x), \quad \Delta X = \text{smat}(\Delta x)$$

be the reconstructed matrices from the corresponding symmetric vectorizations. We can first perform a Cholesky factorization of X

$$X = LL^\top.$$

Then we perform a matrix similarity transformation

$$X + \bar{\alpha}\Delta X \mapsto L^{-1}(X + \bar{\alpha}\Delta X)L^{-\top} = \mathrm{I} + \bar{\alpha}L^{-1}\Delta X L^{-\top},$$

which does not change the eigenvalues of $X + \bar{\alpha}\Delta X$. Therefore, the maximum step size $\hat{\alpha}$ is

$$\hat{\alpha}^{-1} = \lambda_{\max}(-L^{-1}\Delta X L^{-\top}),$$

assuming ΔX is not PSD.

2.4.4 Nondegeneracy

Since the algorithm above is an application of Newton's method to the system of equations (2.26), the key to understand its asymptotic behavior is to analyze the Jacobian (2.27) at the optimal solution. If the Jacobian at the optimal solution is nonsingular, then one can show that the algorithm above has asymptotic quadratic convergence rate.

(Alizadeh et al., 1998) provided a sufficient condition for the Jacobian to be nonsingular, which is called nondegeneracy.

Definition 2.3 (Nondegeneracy)**.** Let (X, y, Z) be a solution to the SDP pair (2.9) and (2.10) and satisfies the KKT optimality conditions (2.22). In this case, X and Z can be simultaneously diagonalized

$$X = Q\text{Diag}(\lambda_1, \ldots, \lambda_n)Q^\top, \quad Z = Q\text{Diag}(\omega_1, \ldots, \omega_n)Q^\top$$

by some orthogonal matrix Q. We assume

$$\lambda_1, \geq \cdots \geq \lambda_n, \quad \omega_1 \leq \ldots \omega_n.$$

Due to $XZ = 0$, we have

$$\lambda_i \omega_i = 0, \quad i = 1, \ldots, n.$$

Let X have rank r with positive eigenvalues $\lambda_1, \ldots, \lambda_r$, and partition $Q = [Q_1, Q_2]$ where the colums of Q_1 are eigenvectors corresponding to $\lambda_1, \ldots, \lambda_r$. We say (X, y, Z) satisfies the strict complementarity and primal and dual nondegeneracy conditions if the following hold:

1. **(Strict Complementarity)** $\text{rank}(Z) = n - r$.
2. **(Primal Nondegeneracy)** The matrices
$$\begin{bmatrix} Q_1^\top A_k Q_1 & Q_1^\top A_k Q_2 \\ Q_2^\top A_k Q_1 & 0 \end{bmatrix}, \quad k = 1, \dots, m \tag{2.33}$$
are linearly independent in $\mathbb{S}^n$.
3. **(Dual Nondegeneracy)** The matrices
$$Q_1^\top A_k Q_1, \quad k = 1, \dots, m \tag{2.34}$$
span the space $\mathbb{S}^r$.

When strict complementarity holds, **primal nondegeneracy implies the dual optimal solution is unique**, and **dual nondegeneracy implies the primal optimal solution is unique**.

Strict complementarity and primal-dual nondegeneracy also immediately imply, recalling $r^\Delta = r(r+1)/2$,
$$r^\Delta \leq m \leq r^\Delta + r(n-r), \tag{2.35}$$
where $r^\Delta \leq m$ is due to (2.34) and $m \leq r^\Delta + r(n-r)$ is due to (2.33).

From (2.35), we can understand that when m is very large, then primal nondegeneracy must fail and the dual solution must not be unique. This is a property we will see later when we study the moment-SOS hierarchy.

2.4.5 Generalization

The AHO Newton direction is obtained by symmetrizing the centrality condition $XZ = \mu \mathrm{I}$ as
$$\frac{1}{2}(XZ + ZX) = \mu \mathrm{I}.$$

It turns out that is not the only way (and also not the most robust way) to symmetrize the centrality condition. A more popular symmetrization is due to (Monteiro, 1997) and (Zhang, 1998), and is typically referred to as the Monteiro-Zhang family (Monteiro, 1998). The basic idea is to symmetrize the centrality condition as
$$\frac{1}{2}(PXZP^{-1} + P^{-\top}XZP^\top) = \mu \mathrm{I}. \tag{2.36}$$
Clearly, the AHO direction is a special case of (2.36) with the choice $P = \mathrm{I}$.

Two other most popular Newton directions are:

- Nesterov-Todd (Todd et al., 1998), where P is obtained by first solving

$$WZW = X,$$

and then setting

$$P = W^{-1/2}.$$

This is the Newton method implemented by SeDuMi.

- HKM, which was proposed and analyzed by multiple authors (Helmberg et al., 1996), (Kojima et al., 1997), computes

$$P = Z^{1/2}.$$

This is the Newton method implemented by SDPT3 and MOSEK.

2.5 Applications

2.5.1 Lyapunov Stability

One of the earliest and most important applications of semidefinite optimization is in the context of dynamical systems and control theory. The main reason is that it is possible to characterize the dynamical properties of a system (e.g., stability, contraction) in terms of algebraic statements such as the feasibility of specific forms of inequalities. We provide a simple example in the case of Lyapunov stability for linear dynamical systems. For a comprehensive overview, see (Boyd et al., 1994).

Consider a discrete-time linear dynamical system

$$x_{k+1} = Ax_k, \quad k = 0, ... \tag{2.37}$$

where $x \in \mathbb{R}^n$ is called the state of the system, and k indexes the time. An important property that one wants to understand about the system (2.37) is whether it is **stable**, i.e., given an initial state x_0, does x_k tend to zero as k tends to infinity.

It is well-known that x_k tends to zero for all initial conditions x_0 if and only if $|\lambda_i(A)| < 1, \forall i = 1, ..., n$, i.e., all the eigenvalues of the transition matrix A lies inside the unit circle.

Equivalently, one can also **certify** the stability of the system if a **Lyapunov function**

$$V(x_k) = x_k^\top P x_k \tag{2.38}$$

is given that satisfies the following matrix inequalities.

Lemma 2.6 (Lyapunov Stability)**.** *Given $A \in \mathbb{R}^{n\times n}$, the following statements are equivalent:*

1. *All eigenvalues of A lie strictly inside the unit circle, i.e., $|\lambda_i(A)| < 1, i = 1, \dots, n$*
2. *There exists a matrix $P \in \mathbb{S}^n$ such that*

$$P \succ 0, \quad A^\top PA - P \prec 0.$$

Proof. $2 \Rightarrow 1$: Let $\lambda \in \mathbb{C}$ and $v \neq 0 \in \mathbb{C}^n$ be a pair of eigenvalue and eigenvector of A such that $Av = \lambda v$. Since $A^\top PA - P \prec 0$, we have

$$v^*(A^\top PA - P)v < 0 \Rightarrow (\lambda v)^* P(\lambda v) - v^* Pv < 0 \Rightarrow (|\lambda|^2 - 1)(v^* Pv) < 0.$$

Since $P \succ 0$, we have $v^* Pv > 0$. Hence, $|\lambda|^2 - 1 < 0$ and $|\lambda| < 1$.

$1 \Rightarrow 2$: Construct

$$P := \sum_{k=0}^{\infty} (A^k)^\top A^k.$$

The sum converges by the condition that $|\lambda_i(A)| < 1, \forall i$ and $P \succ 0$. Then we have

$$A^\top PA - P = \sum_{k=1}^{\infty} (A^k)^\top A^k - \sum_{k=0}^{\infty} (A^k)^\top A^k = -\mathrm{I} \prec 0,$$

proving the result. □

Intuitively, the Lyapunov function (2.38) defines a positive definite energy function. The fact that $P \succ 0$ implies $V(0) = 0$ and $V(x) > 0$ for any x that is nonzero. The condition $A^\top PA - P \prec 0$ guarantees that the system's energy strictly decreases along any possible system trajectory. To see this, write

$$V(x_{k+1}) - V(x_k) = V(Ax_k) - V(x_k) = x_k^\top (A^\top PA - P)x_k < 0, \forall x_k \neq 0.$$

Therefore, the existence of a Lyapunov function guarantees any system trajectory will converge to the origin.

The nice property of constructing such a Lyapunov function is that it goes beyond linear systems and can be similarly generalized to nonlinear systems, where one can compute a stability certificate from semidefinite optimization. See for example this lecture notes.

Control Design. Consider now the case where A is not stable, but we can use a linear **feedback controller** to stabilize the system, i.e.,

$$u_k = Kx_k, \quad x_{k+1} = Ax_k + Bu_k = (A + BK)x_k,$$

where the matrix $K \in \mathbb{R}^{m \times n}$ is the feedback law we wish to design. We wish to design K using convex optimization, in particular semidefinite optimization.

Involving Lemma~2.6, we see that designing K to stabilize $A+BK$ is equivalent to finding

$$P \succ 0, \quad (A + BK)^\top P(A + BK) - P \prec 0. \tag{2.39}$$

Using the Schur complement technique in Proposition 2.1, we know (2.39) is equivalent to

$$\begin{bmatrix} P & (A+BK)^\top P \\ P(A+BK) & P \end{bmatrix} \succ 0.$$

This matrix inequality is, however, not jointly convex in the unknowns (P, A). To address this issue, we can pre-multiply and post-multiply the equation by $\mathrm{BlkDiag}(P^{-1}, P^{-1})$, which leads to

$$\begin{bmatrix} P^{-1} & \\ & P^{-1} \end{bmatrix} \begin{bmatrix} P & (A+BK)^\top P \\ P(A+BK) & P \end{bmatrix} \begin{bmatrix} P^{-1} & \\ & P^{-1} \end{bmatrix} \succ 0 \Leftrightarrow \begin{bmatrix} P^{-1} & P^{-1}(A+BK)^\top \\ (A+BK)P^{-1} & P^{-1} \end{bmatrix} \succ 0. \tag{2.40}$$

Now with a change of variable $Q := P^{-1}$, $Y = KP^{-1}$, we have

$$\begin{bmatrix} Q & QA^\top + Y^\top B^\top \\ AQ + BY & Q \end{bmatrix} \succ 0,$$

which is indeed a linear matrix inequality in the unknowns (Q, Y). Solving the LMI, we can recover

$$P = Q^{-1}, K = YQ^{-1}.$$

2.5.2 Linear Quadratic Regulator

Another (somewhat surprising) application of semidefinite optimization is that it can be used to convexify the direct optimization of the linear quadratic regulator problem (one of the cornerstones in modern optimal control). In fact, such a convex formulation is the key to prove why **policy optimization** works in reinforcement learning, as least in the linear quadratic case (Hu et al., 2023) (Mohammadi et al., 2021).

Consider a continuous-time linear dynamical system

$$\dot{x} = Ax + Bu, \quad x \in \mathbb{R}^n, u \in \mathbb{R}^m, A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}, \tag{2.41}$$

where x denotes the state and u denotes the control. Consider the following optimal control problem

$$\begin{aligned} \min_{u(t)} \quad & \mathbb{E}\left\{ \int_{t=0}^{\infty} x(t)^\top Q x(t) + u(t)^\top R u(t) dt \right\} \\ \text{s.t.} \quad & \dot{x} = Ax + Bu, \quad x(0) \sim \mathcal{N}(0, \Omega) \end{aligned} \tag{2.42}$$

where $Q, R \succ 0$ are known cost matrices, and $x(0)$ is supposed to satisfy the zero-mean Gaussian distribution with covariance $\Omega \succ 0$. In the case where A, B, Q, R are perfectly known to the control designer, and (A, B) is controllable

(to be defined soon), problem (2.42) can be solved exactly and optimally, with the optimal controller being a linear feedback controller of the form

$$u(t) = -Kx(t),$$

where

$$K = R^{-1}B^\top S,$$

with S the unique positive definite solution to the continuous-time **algebraic Riccati equation** (ARE)

$$SA + A^\top S - SBR^{-1}B^\top S + Q = 0.$$

Policy Optimization. The control and reinforcement learning community are interested in whether it is possible to directly find the optimal feedback policy K, i.e., directly solving

$$\begin{array}{ll} \min\limits_{K\in\mathcal{S}} & \mathbb{E}\left\{\int_{t=0}^{\infty} x(t)^\top Qx(t) + u(t)^\top Ru(t)dt\right\} \\ \text{s.t.} & \dot{x} = Ax + Bu, \quad u = -Kx, \quad x(0) \sim \mathcal{N}(0, \Omega). \end{array} \tag{2.43}$$

Note that problem (2.43) is different from the original problem (2.42) in the sense that we assume the knowledge that the optimal controller is linear and directly optimize K. The feasible set $\mathcal{S}$ contains the set of **stabilizing controllers**, i.e.,

$$\mathcal{S} = \{K \in \mathbb{R}^{m\times n} \mid A - BK \text{ is Hurwitz}\}, \tag{2.44}$$

where a matrix A is Hurwitz means $\mathrm{Re}(\lambda_i(A)) < 0$ for all $i = 1, \ldots, n$, i.e., all the eigenvalues of A have strictly negative real parts. When $A-BK$ is Hurwitz, the closed-loop system is stable and the integral in (2.43) will not blow up (so it is sufficient to consider the class of stabilizing controllers).

We will first observe that the direct optimization (2.43) is nonconvex, because the feasible set $\mathcal{S}$ is nonconvex. The following example is adapted from (Fazel et al., 2018).

Example 2.6 (Nonconvex Set of Stabilizing Controllers)**.** Consider the dynamical system (2.41) with

$$A = 0_{3\times 3}, \quad B = \mathrm{I}_3.$$

It is easy to show that both

$$K_1 = \begin{bmatrix} 1 & 0 & -10 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad K_2 = \begin{bmatrix} 1 & -10 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

are stabilizing controllers, as

$$\lambda(A - BK_1) = (-1, -1, -1), \quad \lambda(A - BK_2) = (-1, -1, -1).$$

However, pick

$$K = \frac{K_1 + K_2}{2} = \begin{bmatrix} 1 & -5 & -5 \\ -0.5 & 1 & 0 \\ -0.5 & 0 & 1 \end{bmatrix},$$

we have

$$\lambda(A - BK) = (-3.2361, 1.2361, -1)$$

and K does not stabilize the closed-loop system.

Convex Parameterization with Semidefinite Optimization. We will now show that it is possible to reparameterize problem (2.43) as a semidefinite optimization so it becomes convex. To do so, we note

$$x^\top Qx + u^\top Ru = x^\top Qx + x^\top K^\top RKx = x^\top (Q + K^\top RK)x^\top = \operatorname{tr}((Q + K^\top RK)xx^\top),$$

and therefore the objective of (2.43) can be written as

$$\mathbb{E}\left\{\int_0^\infty \operatorname{tr}((Q + K^\top RK)xx^\top)dt\right\} = \operatorname{tr}\left((Q + K^\top RK)\mathbb{E}\left\{\int_0^\infty xx^\top dt\right\}\right). \tag{2.45}$$

For any given initial condition, the solution of the closed-loop dynamics is given by the matrix exponential

$$x(t) = e^{(A-BK)t}x(0).$$

For the distribution of initial conditions, we have

$$\mathbb{E}\left\{x(t)x(t)^\top\right\} = e^{(A-BK)t}\mathbb{E}\{x(0)x(0)^\top\}e^{(A-BK)^\top t} = e^{(A-BK)t}\Omega e^{(A-BK)^\top t}.$$

The integral of this function, call it X, represents the expected energy of the closed-loop response

$$X = \mathbb{E}\left\{\int_0^\infty xx^\top dt\right\}.$$

For every $K \in \mathcal{S}$ that is stabilizing, X can be computed as the solution to the Lyapunov equation

$$(A - BK)X + X(A - BK)^\top + \Omega = 0. \tag{2.46}$$

As a result, the objective of (2.43) is simplied as

$$\operatorname{tr}((Q + K^\top RK)X),$$

with X solves (2.46). We arrive at an optimization problem

$$\begin{aligned} \min_{X,K} \quad & \operatorname{tr}(QX) + \operatorname{tr}(K^\top RKX) \\ \text{s.t.} \quad & (A - BK)X + X(A - BK)^\top + \Omega = 0, \\ & X \succ 0 \end{aligned} \tag{2.47}$$

which is still nonconvex. We do a change of variable $Y = KX$, so $K = YX^{-1}$ and obtain

$$\begin{aligned} \min_{X,Y} \quad & \mathrm{tr}(QX) + \mathrm{tr}(X^{-1}Y^\top RY) \\ \text{s.t.} \quad & AX - BY + XA^\top - Y^\top B^\top + \Omega = 0 \cdot \\ & X \succ 0 \end{aligned} \tag{2.48}$$

The second term in the objective is still nonconvex, but we can use the Schur complement technique again, and arrive at the final optimization

$$\begin{aligned} \min_{X,Y,Z} \quad & \mathrm{tr}(QX) + \mathrm{tr}(Z) \\ \text{s.t.} \quad & AX - BY + XA^\top - Y^\top B^\top + \Omega = 0 \\ & X \succ 0 \\ & \begin{bmatrix} Z & R^{1/2}Y \\ Y^\top R^{1/2} & X \end{bmatrix} \succeq 0 \end{aligned} \quad . \tag{2.49}$$

Note that the last two matrix inequalities of (2.49) implies, by Schur complement

$$Z - R^{1/2}YX^{-1}Y^\top R^{1/2} \succeq 0 \Rightarrow \mathrm{tr}(Z - R^{1/2}YX^{-1}Y^\top R^{1/2}) \geq 0.$$

Since the optimization is trying to minimize $\mathrm{tr}(Z)$, it will push the equality to hold

$$Z - R^{1/2}YX^{-1}Y^\top R^{1/2} = 0,$$

and thus

$$\mathrm{tr}(Z) = \mathrm{tr}(R^{1/2}YX^{-1}Y^\top R^{1/2}) = \mathrm{tr}(X^{-1}Y^\top RY).$$

Example 2.7 (Convex LQR)**.** For the same linear system as in Example 2.6, let $Q = R = \mathrm{I}_3$.

If we solve the optimal feedback controller K by solving the ARE, we get the optimal controller is $K^\star = \mathrm{I}_3$.

If we implement the SDP in (2.49), we also get $K^\star = \mathrm{I}_3$, confirming the correctness of the convex parameterization. The implementation is shown below and can be found here.

```
%% Show the feasible set is nonconvex
A = zeros(3,3);
B = eye(3);
K1 = [1, 0, -10; -1, 1, 0; 0, 0, 1];
K2 = [1, -10, 0; 0, 1, 0; -1, 0, 1];
K = (K1 + K2)/2;

%% get the groundtruth LQR controller
Q = eye(3); R = eye(3);
```

```
K_lqr = lqr(A,B,Q,R,zeros(3,3));
Omega = eye(3);

%% solve SDP
addpath(genpath('../YALMIP'))
addpath(genpath('../../../mosek'))

X = sdpvar(3,3);
Y = sdpvar(3,3,'full');
Z = sdpvar(3,3);
M = [Z, R*Y; Y'*R, X];
F = [
    A*X - B*Y + X*A' - Y'*B' + Omega == 0;
    M >= 0
];
obj = trace(Q*X) + trace(Z);
optimize(F,obj);
K_sdp = value(Y)*inv(value(X));
```

2.5.3 Domain Adaptation

(Mansour et al., 2009)

Chapter 3

Shor's Semidefinite Relaxation

In this Chapter, we introduce one of the most important and well-known applications of semidefinite optimization, namely its use in the formulation of **convex relaxations** of nonconvex optimization problems.

We will focus on the so-called Shor's semidefinite relaxation (Shor, 1987), which is particularly designed for quadratically constrained quadratic programs (QCQPs). Shor's semidefinite relaxation is relatively easy to formulate and understand, and as we will see later, is essentially the first-order relaxation in the moment-SOS hierarchy.

3.1 Semidefinite Relaxation of QCQPs

Consider a quadratically constrained quadratic program (QCQP):

$$\begin{aligned} f^\star = \min_{x\in\mathbb{R}^n} \quad & x^\top C x \\ \text{s.t.} \quad & x^\top A_i x = b_i, i = 1, \dots, m \end{aligned} \tag{3.1}$$

where $C, A_1, \dots, A_m \in \mathbb{S}^n$ are given symmetric matrices and $b = [b_1, \dots, b_m] \in \mathbb{R}^m$ is a given vector. We assume the problem (3.1) is feasible, solvable, and bounded from below, i.e., $-\infty < f^\star < +\infty$ and is attained. Many practical problems in optimization, engineering, and applied sciences can be formulated as (3.1). For example, problem (3.1) includes binary quadratic optimization (BQP) problems by letting

$$A_i = e_i e_i^\top, i = 1, \dots, n$$

with $e_i \in \mathbb{R}^n$ the standard Euclidean basis vector, in which case the i-th constraint becomes

$$x_i^2 = 1 \Leftrightarrow x_i \in \{+1, -1\}, i = 1, \dots, n.$$

We will discuss a particular type of BQP known as the MAXCUT problem in more details later. From this simple example, since QCQP includes BQP, we know in general the problem (3.1) is nonconvex and it is NP-hard to compute $f^\star$ and find a global optimizer.

3.1.1 Lagrangian Dual Problem

Since the original QCQP is hard to solve in general, we turn to computing a **lower bound** of (3.1). The most natural way to find a lower bound, according to Section 1.3.2, is to derive its Lagrangian dual problem. Towards this goal, we associate a Lagrangian multipler $-y_i$ with each equality constraint and obtain the Lagrangian

$$L(x, y) = x^\top C x - \sum_{i=1}^m y_i (x^\top A_i x - b_i), \quad x \in \mathbb{R}^n, y \in \mathbb{R}^m.$$

The Lagrangian dual, by definition, is

$$\phi(y) = \min_x L(x, y) = \sum_{i=1}^m y_i b_i + \min_{x \in \mathbb{R}^n} x^\top \underbrace{\left(C - \sum_{i=1}^m y_i A_i \right)}_{Z} x.$$

Clearly, if Z is positive semidefinite, then $\min_x x^\top Z x = 0$ (by choosing $x = 0$); otherwise, $\min_x x^\top Z x = -\infty$. Therefore, the dual function is

$$\phi(y) = \begin{cases} \sum_{i=1}^m y_i b_i & \text{if } C - \sum_{i=1}^m y_i A_i \succeq 0 \\ -\infty & \text{otherwise} \end{cases}.$$

The Lagrangian dual problem seeks to maximize y, and hence it will make sure Z is PSD

$$\begin{aligned} d^\star = \max_{y \in \mathbb{R}^m} \quad & b^\top y \\ \text{s.t.} \quad & C - \mathcal{A}^*(y) \succeq 0. \end{aligned} \tag{3.2}$$

By Lagrangian duality, we have

$$d^\star \leq f^\star.$$

Note that problem (3.2) is a convex SDP and can be solved by off-the-shelf solvers.

Another nice property of the dual problem is that it naturally leads to a **certifier**.

Proposition 3.1 (Dual Optimality Certifier)**.** *Let $x_\star$ be a feasible solution to the QCQP* (3.1)*, if there exists $y_\star \in \mathbb{R}^m$ such that*

$$C - \mathcal{A}^*(y_\star) \succeq 0, \quad [C - \mathcal{A}^*(y_\star)]x_\star = 0, \tag{3.3}$$

then $x_\star$ is a global minimizer of (3.1)*, and $y_\star$ is a global maximizer of* (3.2)*.*

Proof. We have zero duality gap

$$\begin{aligned} x_\star^\top C x_\star - b^\top y_\star &= \langle C, x_\star x_\star^\top \rangle - \langle b, y_\star \rangle = \langle C, x_\star x_\star^\top \rangle - \langle \mathcal{A}(x_\star x_\star^\top), y_\star \rangle \\ &= \langle C - \mathcal{A}^*(y_\star), x_\star x_\star^\top \rangle = x_\star^\top [C - \mathcal{A}^*(y_\star)] x_\star = 0. \end{aligned} \tag{3.4}$$

and $(x_\star, y_\star)$ is primal-dual feasible. Therefore, $(x_\star, y_\star)$ is primal-dual optimal. □

The reason why Proposition 3.1 can be quite useful is that it gives a very efficient algorithm to certify global optimality of a candidate (potentially locally) optimal solution $x_\star$. In particular, in several practical applications in computer vision and robotics, the second equation in (3.3) is a linear system of n equations in m variables with $m \leq n$, and hence it has a unique solution. Therefore, one can first solve the linear system and obtain a candidate $y_\star$, and then simply check the PSD condition $C - \mathcal{A}^*(y_\star) \succeq 0$. This leads to optimality certifiers that can run in real time (Garcia-Salguero et al., 2021) (Holmes and Barfoot, 2023).

3.1.2 Dual of the Dual (Bidual)

The dual problem (3.2) should appear familiar to us at this moment – it is simply a standard-form dual SDP (2.10). Therefore, we can write down the SDP dual of the QCQP dual (3.2)

$$\begin{aligned} p^\star = \min_{X \in \mathbb{S}^n} \quad & \langle C, X \rangle \\ \text{s.t.} \quad & \mathcal{A}(X) = b \\ & X \succeq 0 \end{aligned} \tag{3.5}$$

Under the assumption of SDP strong duality (e.g., both (3.5) and (3.2) are strictly feasible), we have

$$p^\star = d^\star \leq f^\star.$$

The weak duality $d^\star \leq f^\star$ can be interpreted using standard Lagrangian duality. How about the weak duality $p^\star \leq f^\star$? It turns out the dual of the dual (bidual) also has a nice interpretation.

We first observe that the original QCQP (3.1) is equivalent to a rank-constrained matrix optimization problem.

Proposition 3.2 (Rank-Constrained Matrix Optimization)**.** *The QCQP* (3.1) *is equivalent to the following rank-constrained matrix optimization*

$$\begin{aligned} f_m^\star = \min_{X\in\mathbb{S}^n} \quad & \langle C, X\rangle \\ \text{s.t.} \quad & \mathcal{A}(X) = b \\ & X \succeq 0 \\ & \text{rank}(X) = 1 \end{aligned} \tag{3.6}$$

in the sense that

1. $f^\star = f_m^\star$
2. *For every optimal solution* $x_\star$ *of the QCQP* (3.1)*,* $X_\star = x_\star x_\star^\top$ *is globally optimal for the matrix optimization* (3.6)
3. *Every optimal solution* $X_\star$ *of the matrix optimization can be factorized as* $X_\star = x_\star x_\star^\top$ *so that* $x_\star$ *is optimal for the QCQP* (3.1)*.*

Proof. We will show that the feasible set of the QCQP (3.1) is the same as the feasible set of (3.6).

Let x be feasible for the QCQP (3.1), we have $X = xx^\top$ must be feasible for (3.6) because $X \succeq 0$ by construction, $\text{rank}(X) = \text{rank}(xx^\top) = \text{rank}(x) = 1$, and

$$x^\top A_i x = b_i \Rightarrow \text{tr}(x^\top A_i x) = b_i \Rightarrow \text{tr}(A_i xx^\top) = b_i \Rightarrow \langle A_i, X\rangle = b_i, \forall i.$$

Conversely, let X be feasible for the matrix optimization (3.6). Since $X \succeq 0$ and $\text{rank}(X) = 1$, $X = xx^\top$ must hold for some $x \in \mathbb{R}^n$. In the meanwhile,

$$\langle A_i, X\rangle = b_i \Rightarrow \langle A_i, xx^\top\rangle = b_i \Rightarrow x^\top A_i x = b_i, \forall i.$$

Therefore, x is feasible for the QCQP (3.1).

Finally, it is easy to observer that

$$\langle C, X\rangle = \langle C, xx^\top\rangle = x^\top C x,$$

and the objective is also the same. □

Since the QCQP is equivalent to the matrix optimization (3.6), one should expect the matrix optimization to be NP-hard in general as well. In fact, this is true due to the nonconvex rank constraint $\text{rank}(X) = 1$. Comparing the nonconvex SDP (3.6) and the convex SDP (3.5), we see the only difference is that we have dropped the nonconvex rank constraint in (3.5) to make it convex, hence the SDP (3.5) is a convex relaxation of the QCQP (3.1) and $p^\star \leq f^\star$.

This convex relaxation perspective also provides a way to certify global optimality, by checking the rank of the optimal solution after solving the convex SDP (3.5).

Proposition 3.3 (Exactness of SDP Relaxation)**.** *Let $X_\star$ be an optimal solution to the SDP* (3.5)*, if* $\operatorname{rank}(X_\star) = 1$*, then $X_\star$ can be factorized as $X_\star = x_\star x_\star^\top$ with $x_\star$ a globally optimal solution to the QCQP* (3.1)*. If so, we say the relaxation* (3.5) *is exact, or tight.*

It is worth noting that, even when the relaxation is exact, i.e., $p^\star = f^\star$, it may not be trivial to numerically certify the exactness, due to several reasons

- If the SDP (3.5) is solved using interior point methods such as MOSEK, then it is well known that they converge to the *maximum-rank solution* (Wolkowicz et al., 2000). This is saying that even if the SDP (3.5) has rank-one solutions, the solvers may not find them. Consider the case where x_1 and x_2 are both optimal solutions to the QCQP (3.1) and the relaxation is exact, it is easy to check that
$$X = \lambda_1 x_1 x_1^\top + \lambda_2 x_2 x_2^\top, \quad \lambda_1, \lambda_2 \geq 0, \lambda_1 + \lambda_2 = 1$$
is globally optimal for the SDP. When $x_1 x_1^\top$ and $x_2 x_2^\top$ are linearly independent, X can have rank equal to two. In this case, interior point methods will not converge to either $x_1 x_1^\top$ or $x_2 x_2^\top$, but will instead find X with some unknown coefficients of λ_1 and λ_2.
- Even if the original QCQP has a unique optimal solution and the relaxation is exact, the SDP solver will converge to a solution that is approximately rank-one (i.e., the second largest sigular value / eigenvalue will still be nonzero) and it may be difficult to draw a conclusion about exactness. Therefore, in practice one can compute a **relative suboptimality gap** by rounding a feasible point to the QCQP from the SDP (it may or may not be easy to round a feasible point), denoted as $\hat{x}$, then compute
$$\hat{f} = \hat{x}^\top C \hat{x},$$
which serves as an upper bound
$$p^\star \leq f^\star \leq \hat{f}.$$
The relative suboptimality gap can be computed as
$$\eta = \frac{|\hat{f} - p^\star|}{1 + |\hat{f}| + |p^\star|}.$$
Clearly, $\eta \approx 0$ certifies exactness of the SDP relaxation.

3.1.3 MAXCUT

We will now study binary quadratic optimization problems, in particular the MAXCUT problem. The original QCQP reads

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & x^\top C x \\ \text{s.t.} \quad & x_i^2 = 1, i = 1, \dots, n. \end{aligned} \tag{3.7}$$

For the MAXCUT problem, a standard formulation is

$$\max_{x_i^2=1} \frac{1}{4} \sum_{i,j} w_{ij}(1 - x_i x_j), \tag{3.8}$$

where $w_{ij} \geq 0$ is the weight of the edge between node i and node j. It is clear that if x_i, x_j have the same sign, then $1 - x_i x_j = 0$, otherwise, $1 - x_i x_j = 2$.

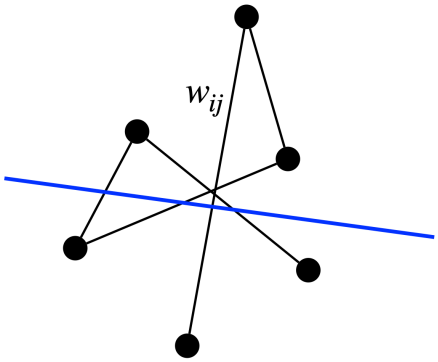

Figure 3.1: MAXCUT seeks to separate the node set of a graph into two disjoint groups such that the separation line cuts as many (weighted) edges as possible. For example, the blue line cuts four edges.

Removing the constant terms in (3.8), it is equivalent to the followng BQP

$$\min_{x_i^2=1} \sum_{i,j} w_{ij} x_i x_j. \tag{3.9}$$

Random Rounding. In general, solving the SDP relaxation of the MAXCUT problem will not produce a certifiably optimal solution. It is therefore interesting to ask if solving the SDP relaxation can produce provably good approximations.

Let X be the optimal solution of the SDP relaxation, and $X = V^\top V$ be a rank-r factorization with $V \in \mathbb{R}^{r \times n}$

$$V = [v_1, \ldots, v_n]$$

and each vector $v_i \in \mathbb{R}^r$. We have $X_{ij} = v_i^\top v_j$. Since $X_{ii} = 1$, the vectors v_i's lie on the unit sphere. Goemans and Williamson (Goemans and Williamson, 1995) proposed to obtain a feasible point to the orignal BQP by first choosing a random unit direction $p \in \mathbb{R}^r$ and then assign

$$x_i = \operatorname{sgn}(p^\top v_i), i = 1, \ldots, n.$$

The expected value of this solution can be written as

$$\mathbb{E}_p \left\{ x^\top C x \right\} = \sum_{i,j} C_{ij} \mathbb{E}_p \left\{ x_i x_j \right\} = \sum_{i,j} C_{ij} \mathbb{E}_p \left\{ \operatorname{sgn}(p^\top x_i) \operatorname{sgn}(p^\top x_j) \right\}.$$

This expectation can be computed using geometric intuition. Consider the plane spanned by v_i and v_j and let θ_{ij} be the angle between them. Then, it is easy to see that the desired expectation is equal to the probability that both points are on the same side of the hyperplane, minus the probability that they are on different sides. These probabilities are $1 - \theta_{ij}/\pi$ and θ_{ij}/π, respectively. Therefore, the expected value of the rounded solution is

$$\sum_{i,j} C_{ij} \left(1 - \frac{2\theta_{ij}}{\pi}\right) = \sum_{i,j} C_{ij} \left(1 - \frac{2}{\pi} \arccos(v_i^\top v_j)\right) = \frac{2}{\pi} \sum_{i,j} C_{ij} \arcsin X_{ij},$$

where we have used

$$\arccos t + \arcsin t = \frac{\pi}{2}.$$

MAXCUT Bound. For the MAXCUT problem, there are constant terms involved in the original cost function, which leads to the expected cut of the rounded solution to be

$$c_{\text{expected}} = \frac{1}{4} \sum_{i,j} w_{ij} \left(1 - \frac{2}{\pi} \arcsin X_{ij}\right) = \frac{1}{4} \frac{2}{\pi} \sum_{ij} w_{ij} \arccos X_{ij}.$$

On the other hand, the optimal value of the SDP relaxation produces an upper bound on the true MAXCUT

$$c_{\text{ub}} = \frac{1}{4} \sum_{i,j} w_{ij}(1 - X_{ij}).$$

We have

$$c_{\text{expected}} \leq c_{\text{MAXCUT}} \leq c_{\text{ub}}.$$

We want to find the maximum possible α such that

$$\alpha c_{\text{ub}} \leq c_{\text{expected}} \leq c_{\text{MAXCUT}} \leq c_{\text{ub}},$$

so that α acts to be the best approximation ratio. To find such α, we need to find the maximum α such that

$$\alpha(1 - t) \leq \frac{2}{\pi} \arccos(t), \forall t \in [-1, 1].$$

The best possible α is 0.878, see Fig. 3.2.

3.2 Certifiably Optimal Rotation Averaging

Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V} = [N]$ and edge set $\mathcal{E} = \{(i, j) \mid i, j \in \mathcal{V}\}$. Each node i is associated with an unknown rotation matrix $R_i \in \text{SO}(3)$, and each edge is associated with a relative rotation

$$\tilde{R}_{ij} = R_i^\top R_j \cdot R_\epsilon, \tag{3.10}$$

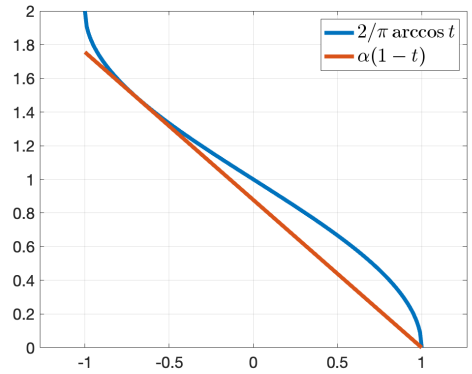

Figure 3.2: Best approximation ratio.

that measures the relative rotation between R_i and R_j, up to some small noise corruption $R_\epsilon \in \mathrm{SO}(3)$. See Fig. 3.3.

The goal of (multiple) rotation averaging is to estimate the absolute rotations $\{R_i\}_{i=1}^N$ given the noisy relative rotation measurements on the edges $\{\tilde{R}_{ij}\}_{(i,j)\in\mathcal{E}}$. This problem is also known as rotation synchronization and it finds applications in computer vision (Eriksson et al., 2018), robotics (Rosen et al., 2019), and medical imaging (Wang and Singer, 2013).

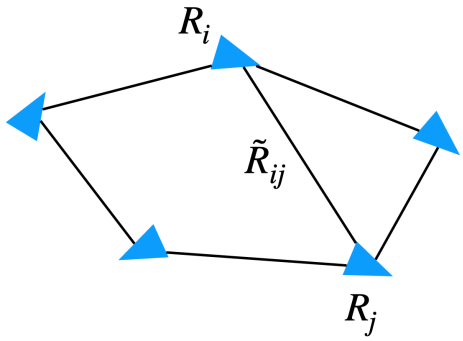

Figure 3.3: Rotation Averaging.

To synchronize the absolute rotations from relative measurements, it is common practice to formulate the following optimization problem

$$\min_{R_i\in\mathrm{SO}(3),i\in\mathcal{V}} \sum_{(i,j)\in\mathcal{E}} \|\tilde{R}_{ij} - R_i^\top R_j\|_{\mathrm{F}}^2, \tag{3.11}$$

to seek the best absolute rotations that fit the relative measurements according

to the generative model (3.10). It can be shown that when the noise R_ϵ satisfies a Langevin distribution, then problem (3.11) returns the maximum likelihood estimator. Even when the noise distribution is not Langevin, problem (3.11) often produces accurate estimates.

QCQP Formulation. We will first simplify problem (3.11) as a QCQP. Note that the objective is equivalent to

$$\sum_{(i,j)\in\mathcal{E}} \operatorname{tr}\left((\tilde{R}_{ij} - R_i^\top R_j)^\top (\tilde{R}_{ij} - R_i^\top R_j)\right) = \sum_{(i,j)\in\mathcal{E}} \operatorname{tr}\left(2\mathrm{I}_3 - 2\tilde{R}_{ij}^\top R_i^\top R_j\right).$$

Therefore, problem (3.11) is equivalent to

$$\min_{R_i\in\mathrm{SO}(3), i\in\mathcal{V}} -2 \sum_{(i,j)\in\mathcal{E}} \operatorname{tr}(\tilde{R}_{ij}^\top R_i^\top R_j). \tag{3.12}$$

This is a QCQP because the objective is quadratic, and SO(3) can be described by quadratic equality constraints.

Matrix Formulation. Let

$$R = \begin{bmatrix} R_1^\top \\ \vdots \\ R_N^\top \end{bmatrix} \in \mathrm{SO}(3)^N, \quad RR^\top = \begin{bmatrix} R_1^\top R_1 & R_1^\top R_2 & \cdots & R_1^\top R_N \\ R_2^\top R_1 & R_2^\top R_2 & \cdots & R_2^\top R_N \\ \vdots & \vdots & \ddots & \vdots \\ R_N^\top R_1 & R_N^\top R_2 & \cdots & R_N^\top R_N \end{bmatrix} \in \mathbb{S}_+^{3N}$$

and

$$\tilde{R} = \begin{bmatrix} 0 & \tilde{R}_{12} & \cdots & \tilde{R}_{1N} \\ \tilde{R}_{12}^\top & 0 & \cdots & \tilde{R}_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{R}_{1N}^\top & \tilde{R}_{2N}^\top & \cdots & 0 \end{bmatrix} \in \mathbb{S}^{3N} \tag{3.13}$$

Then problem (3.12) can be compactly written as

$$\min_{R\in\mathrm{SO}(3)^N} -\langle \tilde{R}, RR^\top \rangle \tag{3.14}$$

Semidefinite Relaxation. Problem (3.14) is nonconvex, so we apply semidefinite relaxation. Observe that, because $R_i \in \mathrm{SO}(3)$, we have

$$R_i^\top R_i = \mathrm{I}_3, \forall i = 1, \dots, N.$$

Therefore, the diagonal blocks of $RR^\top$ are all 3×3 identity matrices. We have also that $RR^\top \succeq 0$ by construction. Therefore, the following SDP is a convex relaxation of problem (3.14)

$$\begin{aligned} \min_{X\in\mathbb{S}^{3N}} \quad & -\langle \tilde{R}, RR^\top \rangle \\ \text{s.t.} \quad & X \succeq 0 \\ & [X]_{ii} = \mathrm{I}_3, \quad i = 1, \dots, N \end{aligned} \tag{3.15}$$

where the last constraint in (3.15) enforces all diagonal blocks to be identity matrices.

How powerful is this SDP relaxation? You may think that it will be quite loose (and hence not very useful) because we have dropped many nonconvex constraints from the original QCQP to the convex SDP (e.g., $\text{rank}(X) = 3$, $R_i \in \text{SO}(3)$ but we only used $R_i \in \text{O}(3)$), but empirically it is almost always exact.

Example 3.1 (Rotation Averaging)**.** I will generate a fully connected graph $\mathcal{G}$ with N nodes.

- For each node i, I associate a random 3D rotation matrix R_i. For the first node, I always let $R_1 = \text{I}_3$.
- For each edge (i, j), I generate a noisy measurement

  $$\tilde{R}_{ij} = R_i^\top R_j \cdot R_\epsilon,$$

  where R_ϵ is a rotation matrix with a random rotation axis and a rotation angle uniformally distributed between 0 and β degrees.

After this graph is generated, I form the $\tilde{R}$ matrix in (3.13) and solve the SDP (3.15). This can be easily programmed in Matlab:

```
X = sdpvar(3*n,3*n);
F = [X >= 0];
for i = 1:n
    F = [F,
        X(blkIndices(i,3),blkIndices(i,3)) == eye(3)];
end
obj = trace(-Rtld*X);
optimize(F,obj);
Xval = value(X);
f_sdp = value(obj);
```

Note that `f_sdp` will be a lower bound.

Let $X_\star$ be the optimal solution of the SDP. We know that, if the SDP relaxation is tight, then $X_\star$ will look like $RR^\top$. Because $R_1 = \text{I}_3$, we can directly read off the optimal rotation estimations from the first row of blocks. However, if the relaxation is not tight, the blocks there will not be valid rotation matrices, and we can perform a projection onto SO(3). Using the estimated rotations, I can compute `f_est`, which is an upper bound to the true global optimum.

```
R_est = [];
R_errs = [];
for i = 1:n
    if i == 1
        Ri = eye(3);
    else
        Ri = project2SO3(Xval(1:3,blkIndices(i,3)));
    end
    R_errs = [R_errs, getAngularError(Ri, R_gt(blkIndices(i,3),:))];
    R_est = [R_est, Ri];
end
X_est = R_est'*R_est;
f_est = trace(-Rtld*X_est);
```

With `f_est` and `f_sdp`, an upper bound and a lower bound, I can compute the relative suboptimality gap η. If $\eta = 0$, then it certifies global optimality.

How does this work?

For $N = 30$ and $\beta = 10$ (small noise), I got $\eta = 6.26 \times 10^{-13}$. Fig. 3.4 plots the rotation estimation errors at each node compared to the groundtruth rotations.

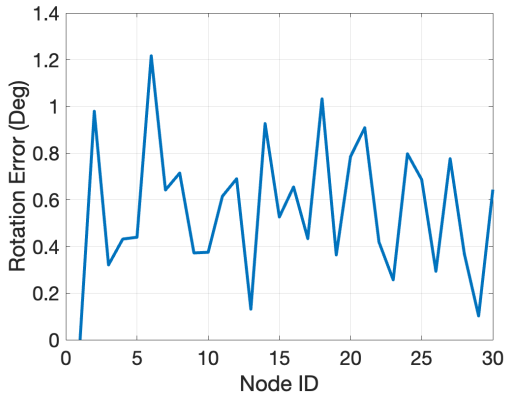

Figure 3.4: Rotation estimation errors, 30 nodes, noise bound 10 degrees.

What if I increase the noise bound to $\beta = 60$? It turns out the relaxation is still exact with $\eta = 6.84 \times 10^{-10}$! Fig. 3.5 plots the rotation estimation errors at each node compared to the groundtruth rotations.

What if I set $\beta = 120$? The relaxation still remains exact with $\eta = 9.5 \times 10^{-10}$. Fig. 3.6 plots the rotation estimation errors at each node compared to the groundtruth rotations. Observe that even when the rotation estimates have large errors, they are still the certifiably optimal estimates.

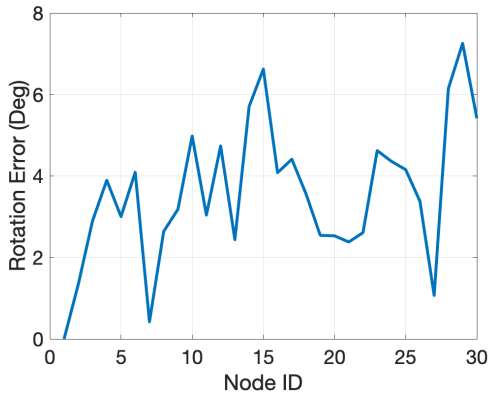

Figure 3.5: Rotation estimation errors, 30 nodes, noise bound 60 degrees.

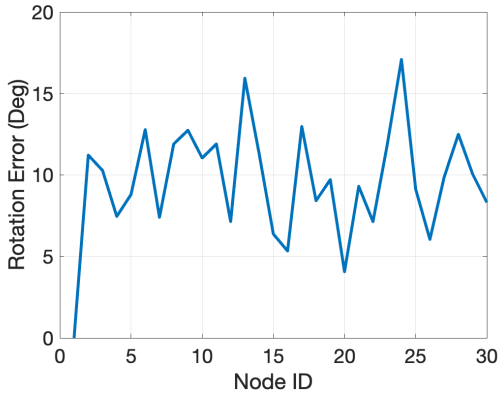

Figure 3.6: Rotation estimation errors, 30 nodes, noise bound 120 degrees.

Play with the code yourself to appreciate the power of this simple SDP relaxation. You can, for example, increase the number of nodes N. What if you make the graph sparse (e.g., fewer edges but still a connected graph)?

3.2.1 Dual Optimality Certifier

3.3 Stretch to High-Degree Polynomial Optimization

We have seen that Shor's semidefinite relaxation for QCQPs can be derived both using Lagrangian duality, or simply by dropping a rank constraint, both of which are straightforward to understand. When applied to the MAXCUT problem, it produces a provably good approximation ratio. When applied to the rotation averaging problem, it directly gives us the optimal solution without any approximation, and the optimality comes with a certificate.

However, not every optimization problem is a QCQP, right? Is it possible to generalize Shor's semidefinite relaxation to higher-degree polynomial optimization problems? As we will see in the next Chapters, the moment and sums-of-squares (SOS) hierarchy delivers the perfect and principled generalization.

Before going to the moment-SOS hierarchy, let me give you an example of a quartic (degree-4) optimization problem, for which we can still use Shor's relaxation, albeit with some (in my opinion, not so elegant) mathematical massage. This example in fact is from a recent paper in computer vision (Briales et al., 2018).

Two-view Geometry. Consider the problem of estimating the motion of a camera from two views illustrated in Fig. 3.7. Let C_1 and C_2 be two cameras (or the same camera but in two different positions) observing the same 3D point $p \in \mathbb{R}^3$. The 3D point will be observed by C_1 and C_2 via its 2D projections on the image plane, respectively. Let $f_1 \in \mathbb{R}^3$ and $f_2 \in \mathbb{R}^3$ be the unit-length bearing vector that emanates from the camera centers to the 2D projections in two cameras, respectively. Our goal is to estimate the relative rotation and translation between the two cameras C_1 and C_2, denoted by $R \in \mathrm{SO}(3)$ and $t \in \mathbb{R}^3$. The pair of bearing vectors (f_1, f_2) is typically known as a **correspondence**, or a **match** in computer vision.

It turns out only having one correspondence is insufficient to recover (R, t), and we need at least 5 such correspondences (Nistér, 2004). See Fig. 3.8 for an example I adapted from Mathworks.

Consequently, to formally state our problem, we are given a set of N correspondences (these correspondences are typically detected by neural networks today (Wang et al., 2020))

$$\{f_{1,i}, f_{2,i}\}_{i=1}^{N}$$

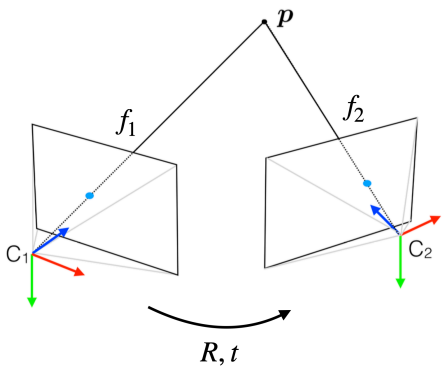

Figure 3.7: Two-view Geometry.

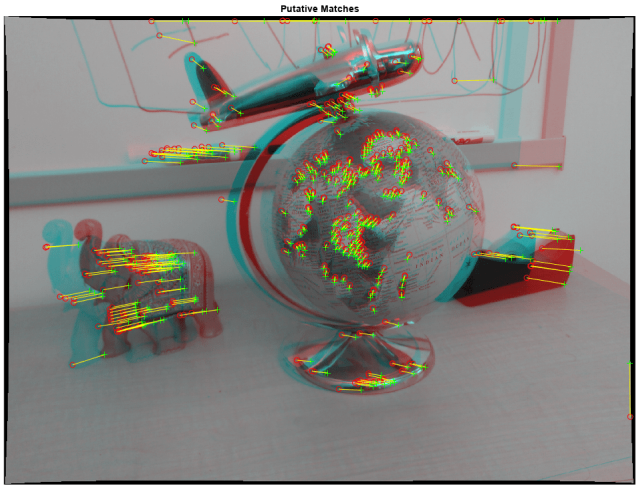

Figure 3.8: A real two-view motion estimation example. Copyright: Mathworks.

between two images taken by two cameras, and our goal is to estimate the relative motion (R, t).

Epipolar Constraint. When the correspondences are noise-free, it is known that they must satisfy the following epipolar constraint (Hartley and Zisserman, 2003)

$$f_{2,i}^\top [t]_\times R f_{1,i} = 0, \quad \forall i = 1, \dots, N, \tag{3.16}$$

where

$$[t]_\times := \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

is a linear map such that $t \times v = [t]_\times v$ for any $v \in \mathbb{R}^3$ and $\times$ denotes cross product in 3D.

Nonlinear Least Squares. Since the correspondences are detected by neural networks, they will be noisy and the epipolar constraint (3.16) will not be perfectly satisfied. Therefore, we formulate the following optimization problem to seek the best estimate that minimize the sum of the squared violations of (3.16)

$$\min_{R \in \mathrm{SO}(3), t \in \mathcal{S}^2} \sum_{i=1}^{N} (f_{2,i}^\top [t]_\times R f_{1,i})^2 \tag{3.17}$$

Note that I have asked $t \in \mathcal{S}^2$ to lie on the unit sphere, why?

Problem (3.17) is not a QCQP anymore, because its objective is a degree-4 polynomial. However, all the constraints of (3.17) are quadratic equalities and inequalities. To see this, note that $t \in \mathcal{S}^2$ can be written as

$$1 - t^\top t = 0, \tag{3.18}$$

which is a quadratic polynomial equality. The constraint $R \in \mathrm{SO}(3)$ can also be written as quadratic equalities. Let

$$R = [c_1, c_2, c_3]$$

where c_i is the i-the column. Then $R \in \mathrm{SO}(3)$ is equivalent to

$$\begin{aligned} c_i^\top c_i - 1 = 0, &\quad i = 1, 2, 3 \\ c_i^\top c_j = 0, &\quad (i, j) \in \{(1, 2), (2, 3), (3, 1)\} \\ c_i \times c_j = c_k, &\quad (i, j, k) \in \{(1, 2, 3), (2, 3, 1), (3, 1, 2)\}. \end{aligned} \tag{3.19}$$

all of which are quadratic polynomial equalities.

Let $r = \mathrm{vec}(R)$, $x = [r^\top, t^\top]^\top \in \mathbb{R}^{12}$, we will collectively call all the constraints in (3.18) and (3.19) as

$$h_k(x) = 0, k = 1, \dots, l,$$

with $l = 16$.

Semidefinite Relaxation. Clearly, we cannot directly apply Shor's semidefinite relaxation, because $X = xx^\top$ only contains monomials in x of degree up to 2, but our objective function is degree 4 – this matrix variable is not powerful enough.

To fix this issue, a natural idea is to create a larger matrix variable

$$v = \begin{bmatrix} 1 \\ r \\ t \\ t_1 r \\ t_2 r \\ t_3 r \end{bmatrix} \in \mathbb{R}^{40}, \quad X = vv^\top = \begin{bmatrix} 1 & * & * & * & * & * \\ r & rr^\top & * & * & * & * \\ t & tr^\top & tt^\top & * & * & * \\ t_1 r & t_1 rr^\top & t_1 rt^\top & t_1^2 rr^\top & * & * \\ t_2 r & t_2 rr^\top & t_2 rt^\top & t_1 t_2 rr^\top & t_2^2 rr^\top & * \\ t_3 r & t_3 rr^\top & t_3 rt^\top & t_3 t_1 rr^\top & t_3 t_2 rr^\top & t_3^2 rr^\top \end{bmatrix} \in \mathbb{S}^{40}_+ \tag{3.20}$$

Note that now X has degree-4 monomials, which allows me to write the objective of the original problem (3.17) as

$$\langle C, X \rangle$$

with a suitable constant matrix $C \in \mathbb{S}^{40}$.

I can also write all the original constraints $h_k(x) = 0, k = 1, \dots, l$ as

$$\langle A_k, X \rangle = 0, k = 1, \dots, l,$$

plus an additional constraint

$$\langle A_0, X \rangle = 1,$$

where A_0 is all zero except its top-left entry is equal to 1.

This seems to be all we need for Shor's relaxation, will this work?

"Redundant" Constraints. In the original paper (Briales et al., 2018), the authors found that we are missing some important constraints that we can add to the convex SDP. For example, in the matrix X, we have

$$t_1^2 rr^\top + t_2^2 rr^\top + t_3^2 rr^\top = (t_1^2 + t_2^2 + t_3^2) rr^\top = rr^\top,$$

which gives us additional linear constraints on the entries of X (for free)! Essentially, these constraints are generated by multiplying the original constraints $h_k(x)$ by suitable monomials:

$$h_k(x) = 0 \Rightarrow h_k(x) \cdot \lambda(x) = 0,$$

where $\lambda(x)$ is a monomial such that all the monomials of $h_k(x) \cdot \lambda(x)$ appear in the big matrix X (so that the resulting equality constraint can still be written as a linear equality on X). The authors of (Briales et al., 2018) enumerated all such constraints (by hand) and added them to the SDP relaxation, which led to the relaxation going from loose to tight/exact.

(Briales et al., 2018) called these constraints "redundant", are they? We will revisit this after we study the moment-SOS hierarchy!

Chapter 4

Sums of Squares Relaxation

4.1 Basic Algebraic Geometry

We first review several basic concepts in algebraic geometry. We refer to standard textbooks for a more comprehensive treatment (Bochnak et al., 2013), (Cox et al., 2013), (Dummit and Foote, 2004), (Lang, 2012).

4.1.1 Groups, Rings, Fields

Definition 4.1 (Group)**.** A **group** consists of a set G and a binary operation "$\cdot$" defined on G that satisfies the following conditions:

1. Associative: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$, for all $a, b, c \in G$.
2. Identity: there exists $1 \in G$ such that $1 \cdot a = a \cdot 1 = a$, for all $a \in G$.
3. Inverse: Given $a \in G$, there exists $b \in G$ such that $a \cdot b = b \cdot a = 1$.

For example, the integers $\mathbb{Z}$ form a group under addition, but not under multiplication; the set $GL(n, \mathbb{R})$ that contains nonsingular $n \times n$ matrices forms a group under the usual matrix multiplication. Another example is the set of rotation matrices $\mathrm{SO}(d) := \{R \in \mathbb{R}^{d \times d} \mid RR^\top = \mathrm{I}_d, \det(R) = +1\}$.

In a group we only have one binary operation ("multiplication"). We will introduce another operation ("addition").

Definition 4.2 (Commutative Ring)**.** A **commutative ring** (with identity) consists of a set S and two binary operations "$\cdot$" and "$+$" defined on S that satisfy the following conditions

1. Associative: $(a+b)+c = a+(b+c)$, and $(a \cdot b) \cdot c = a \cdot (b \cdot c)$, for all $a, b, c \in S$.

2. Commutative: $a+b = b+a$ and $a \cdot b = b \cdot a$, for all $a, b \in S$.

3. Distributive: $a \cdot (b+c) = a \cdot b + a \cdot c$ for all $a, b, c \in S$.

4. Identities: there exist $0, 1 \in S$ such that $a + 0 = a \cdot 1 = a$, for all $a \in S$.

5. Additive inverse: given $a \in S$, there exists $b \in S$ such that $a + b = 0$.

A simple example of a ring is the set of integers $\mathbb{Z}$ under the usual addition and multiplication.

If we add a requirement for the existence of multiplicative inverse, we obtain a field.

Definition 4.3 (Field)**.** A **field** consists of a set S and two binary operations "$\cdot$" and "$+$" defined on S that satisfy the following conditions

1. Associative: $(a+b)+c = a+(b+c)$, and $(a \cdot b) \cdot c = a \cdot (b \cdot c)$, for all $a, b, c \in S$.

2. Commutative: $a+b = b+a$ and $a \cdot b = b \cdot a$, for all $a, b \in S$.

3. Distributive: $a \cdot (b+c) = a \cdot b + a \cdot c$ for all $a, b, c \in S$.

4. Identities: there exist $0, 1 \in S$, where $0 \neq 1$, such that $a + 0 = a \cdot 1 = a$, for all $a \in S$.

5. Additive inverse: given $a \in S$, there exists $b \in S$ such that $a + b = 0$.

6. Multiplicative inverse: given $a \in S$ and $a \neq 0$, there exists $c \in S$ such that $a \cdot c = 1$.

Any field is obvious a commutative ring. Some commonly used fields are the rationals $\mathbb{Q}$, the reals $\mathbb{R}$, and the complex numbers $\mathbb{C}$. Another important field is given by $k(x_1, \ldots, x_n)$, the set of rational functions with coefficients in the field k, with the natural operations.

4.1.2 Polynomials, Ideals, and Varieties

We will use $\mathbb{F} = \mathbb{R}$ or $\mathbb{C}$ to denote the field of real or complex numbers from now on. Let $x_1, \ldots, x_n$ be indeterminates, we can define a polynomial.

Definition 4.4 (Polynomial)**.** A polynomial f in $x_1, \dots, x_n$ with coefficients in a field $\mathbb{F}$ is a finite linear combination of monomials:

$$f = \sum_{\alpha} c_\alpha x^\alpha = \sum_{\alpha} c_\alpha x_1^{\alpha_1} \cdots x_n^{\alpha_n}, \quad c_\alpha \in \mathbb{F},$$

where the sum is over a finite number of n-tuples (exponents) $\alpha = (\alpha_1, \dots, \alpha_n)$, $\alpha_i \in \mathbb{N}$. The set of all polynomials in x with coefficients in $\mathbb{F}$ is denoted $\mathbb{F}[x]$ or $\mathbb{F}[x_1, \dots, x_n]$.

The **degree** of a monomial is the sum of its exponents:

$$\deg(x^\alpha) = \deg(x_1^{\alpha_1} \cdots x_n^{\alpha_n}) = \sum_{i=1}^{n} \alpha_i.$$

The degree of a polynomial is the maximum degree of its monomials:

$$\deg(f) = \max_{\alpha} \deg(x^\alpha).$$

It is clear that $\mathbb{F}[x]$ is a commutative ring with the 0 and 1 identities.

A **form** is a polynomial where all the monomials have the same degree. It is also called a **homogeneous polynomial**. For example,

$$f = 2x_1^2 + x_1 x_2 + x_2^2$$

is a form of degree 2. A homogeneous polynomial of degree d satisfies

$$f(\lambda x_1, \dots, \lambda x_n) = \lambda^d f(x_1, \dots, x_n).$$

A polynomial in n variables of degree d has

$$s(n, d) = \binom{n+d}{d}$$

coefficients. Let $\mathbb{F}[x]_d$ be the set of polynomials in n variables of degree d, then any $f \in \mathbb{F}[x]_d$ can be written as

$$f = c^\top [x]_d, \quad c \in \mathbb{F}^{s(n,d)}$$

with $[x]_d$ the **standard monomial basis** in x of degree up to d. For example, when $x = (x_1, x_2)$, then

$$[x]_2 = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}.$$

Let G be a set of polynomials, and $\mathbb{F}[G]$ denote the set of all polynomials that can be written as

$$\sum_{\alpha} c_\alpha g_1^{\alpha_1} \cdots g_k^{\alpha_k}, \quad c_\alpha \in \mathbb{F}, g_1, \ldots, g_k \in G$$

with finitely many nonzero coefficients. The polynomials in G are called **generators** and G is called a **generator set** for $\mathbb{F}[G]$. Clearly, the set

$$G = \{1, x_1, \ldots, x_n\}$$

is a generator set for $\mathbb{F}[x]$.

We consider next ideals, which are subrings with an "absorbent" property.

Definition 4.5 (Ideal)**.** Let R be a commutative ring. A subset $I \subset R$ is an ideal if it satisfies

1. $0 \in I$.
2. If $a, b \in I$, then $a + b \in I$.
3. If $a \in I$ and $b \in R$, then $a \cdot b \in I$.

A simple example of an ideal is the set of even integers, considered as a subset of the integer ring $\mathbb{Z}$. If the ideal I contains the multiplicative identity "1", then $I = R$. For a tuple $h = (h_1, \ldots, h_s)$ of polynomials in $\mathbb{F}[x]$, Ideal$[h]$:= Ideal$[h_1, \ldots, h_m]$ denotes the smallest ideal containing h, or equivalently

$$\text{Ideal}[h] = h_1 \cdot \mathbb{F}[x] + \cdots + h_s \cdot \mathbb{F}[x].$$

The set Ideal$[h]$ is called the ideal generated by h. Every ideal of $\mathbb{F}[x]$ is generated by finitely many polynomials, i.e., every ideal is **finitely generated**.

Theorem 4.1 (Hilbert Basis Theorem)**.** *For every ideal $I \subseteq \mathbb{F}[x]$, there exist finitely many polynomials $g_1, \ldots, g_m \in I$ such that $I =$* Ideal$[g_1, \ldots, g_m]$.

We define the concept of an **algebraic variety** as the zero set of a set of polynomial equations.

Definition 4.6 (Affine Variety)**.** Let $f_1, \ldots, f_s \in \mathbb{F}[x_1, \ldots, x_n]$ and $x = (x_1, \ldots, x_n)$, and the set V be

$$V_{\mathbb{F}}(f_1, \ldots, f_s) = \{x \in \mathbb{F}^n \mid f_i(x) = 0, i = 1, \ldots, s\}.$$

We call $V_{\mathbb{F}}(f_1, \ldots, f_s)$ the affine variety defined by $f_1, \ldots, f_s$.

Similarly, let $I \subset \mathbb{F}[x]$ be an ideal, we denote its zero set as

$$V_{\mathbb{F}}(I) = \{x \in \mathbb{F}^n \mid f(x) = 0, \forall f \in I\}.$$

Since the ideal is finitely generated, we have

$$V_{\mathbb{F}}(I) = V_{\mathbb{F}}(g_1, \dots, g_m),$$

where $g_1, \dots, g_m$ are the generators of I.

The set of polynomials that vanish in a given variety, i.e.,

$$I(V) = \{f \in \mathbb{F}[x] \mid f(x) = 0, \forall x \in V\},$$

is an ideal, called the **vanishing ideal** of V.

We define the **radical** of an ideal.

Definition 4.7 (Radical)**.** Let $I \subset \mathbb{F}[x]$ be an ideal. The radical of I, denoted $\sqrt{I}$, is the set

$$\sqrt{I} := \{f \mid f^k \in I \text{ for some integer } k\}.$$

It is clear that $I \subset \sqrt{I}$, and it can be shown that $\sqrt{I}$ is also a polynomial ideal.

Given an ideal I and $V_{\mathbb{C}}(I)$, it is clear that any $f \in I$ vanishes on $V_{\mathbb{C}}(I)$, that is

$$I \subseteq I(V_{\mathbb{C}}(I)).$$

However, not all polynomials that vanish on $V_{\mathbb{C}}(I)$ belong to I.

Theorem 4.2 (Hilbert's Nullstellensatz)**.** *Let $I \subseteq \mathbb{C}[x]$ be an ideal.*

- *(Weak Nullstellensatz) If $V_{\mathbb{C}}(I) = \emptyset$, then $1 \in I$.*
- *(Strong Nullstellensatz) For $f \in \mathbb{C}[x]$, if $f(u) = 0$ for all $u \in V_{\mathbb{C}}(I)$, then $f^k \in I$ for some integer $k \geq 1$, i.e., $I(V_{\mathbb{C}}(I)) = \sqrt{I}$.*

Let us see an example of Hilbert's Nullstellensatz.

Example 4.1 (Hilbert's Nullstellensatz)**.** Consider the ideal

$$I = \text{Ideal}[x_1^2 x_2^3, x_1 x_2^4].$$

The affine variety defined by I is

$$V = V_{\mathbb{C}}(I) = \{(x_1, x_2) \in \mathbb{C}^2 \mid x_1^2 x_2^3 = 0, x_1 x_2^4 = 0\}.$$

It is easy to see that the variety is the union of the two coordinate axes $x_1 = 0$ and $x_2 = 0$.

Therefore, the polynomial $x_1 x_2$ vanishes on the variety V, but $x_1 x_2$ does not belong to the ideal I (because the degree is lower).

We claim that

$$\sqrt{I} = \text{Ideal}[x_1 x_2].$$

Indeed, $x_1 x_2 \in \sqrt{I}$ because

$$(x_1 x_2)^3 = x_1^3 x_2^3 = x_1 \cdot x_1^2 x_3^3 \in I.$$

Conversely, if $f \in \sqrt{I}$, then $f^k \in I$ for some integer k, i.e.,

$$f^k = x_1^2 x_2^3 p(x) + x_1 x_2^4 q(x) = x_1 x_2 r(x)$$

which implies $f^k \in \text{Ideal}[x_1 x_2]$. Therefore, all the polynomials that vanish on the variety V can be generated by $x_1 x_2$.

As another example, consider the ideal generated by a single univariate polynomial

$$I = \text{Ideal}[f], \quad f = \prod_{i=1}^{s} (x - a_i)^{n_i}$$

with $a_i \neq a_j$ for $i \neq j$ the unique roots of the polynomial f. Clearly, the affine variety defined by I is the set of unique roots:

$$V = V_{\mathbb{C}}(I) = \{a_1, \ldots, a_s\}.$$

The vanishing ideal of V is

$$\sqrt{I} = \text{Ideal}[(x - a_1) \cdots (x - a_s)].$$

4.1.3 Gröbner Bases

A polynomial ideal can have different "descriptions" in terms of its generators. We will now focus on a particular type of description that is better than the others.

Let's first define a monomial ordering.

Definition 4.8 (Monomial Ordering)**.** A monomial ordering on $\mathbb{C}[x]$ is a binary relation $\succ$ on $\mathbb{N}^n$, i.e., the exponents of the monomials, that satisfies

1. The relation $\succ$ is a total ordering, i.e., given any two monomials x^α, x^β, we must have either $x^\alpha \succ x^\beta$, or $x^\beta \succ x^\alpha$, or $x^\alpha = x^\beta$.
2. If $x^\alpha \succ x^\beta$, then $x^\alpha \cdot x^\gamma \succ x^\beta \cdot x^\gamma$ for any monomial x^γ.
3. The relation $\succ$ is well ordered, i.e., every nonempty set has a smallest element under $\succ$.

There are several monomial orderings of interest in computational algebra.

- **Lexicographic** ("dictionary"). Here $\alpha \succ_{\text{lex}} \beta$ if the left-most nonzero entry of $\alpha - \beta$ is positive. Note that a particular ordering of the variables is assumed.
- **Graded lexicographic**. Sort first by total degree, then Lexicographic, i.e., $\alpha \succ_{\text{grlex}} \beta$ if $|\alpha| > |\beta|$, or $|\alpha| = |\beta|$ and $\alpha \succ_{\text{lex}} \beta$.
- **Graded reverse lexicographic**. Here $\alpha \succ_{\text{grevlex}} \beta$ if $|\alpha| > |\beta|$ or $|\alpha| = |\beta|$ but the right-most nonzero entry of $\alpha - \beta$ is negative. This ordering, although somewhat nonintuitive, has some desirable computational properties.

Example 4.2 (Monomial Ordering)**.** Consider the polynomial ring $\mathbb{C}[x, y]$ in two variables. With the lexicographic ordering $\prec_{\text{lex}}$, we have

$$1 \prec_{\text{lex}} y \prec_{\text{lex}} y^2 \prec_{\text{lex}} \cdots \prec_{\text{lex}} x \prec_{\text{lex}} xy \prec_{\text{lex}} xy^2 \prec_{\text{lex}} \cdots \prec_{\text{lex}} x^2 \prec_{\text{lex}} x^2y \prec_{\text{lex}} x^2y^2 \prec_{\text{lex}} \cdots.$$

For the other two orderings $\prec_{\text{grlex}}$ and $\prec_{\text{grevlex}}$, which in the case of two variables are the same, we have

$$1 \prec y \prec x \prec y^2 \prec xy \prec x^2 \prec y^3 \prec xy^2 \prec x^2y \prec x^3 \prec \cdots.$$

As another example, consider the monomials $\alpha = x^3y^2z^8$ and $\beta = x^2y^9z^2$. If the variables are ordered as (x, y, z), then

$$\alpha \succ_{\text{lex}} \beta, \quad \alpha \succ_{\text{grlex}} \beta, \quad \alpha \prec_{\text{grevlex}} \beta.$$

Note that $x \succ y \succ z$ for all three orderings.

We define a monomial ideal.

Definition 4.9 (Monomial Ideal)**.** A monomial ideal is a polynomial ideal that can be generated by monomials.

What are the possible monomials that belong to a given monomial ideal? Since $x^\alpha \in I \Rightarrow x^{\alpha+\beta} \in I$ for any $\beta \geq 0$, we have that these sets are "closed upwards".

A polynomial belongs to a monomial ideal I if and only if its terms are in I.

Similarly, we have that every monomial ideal is finitely generated.

Theorem 4.3 (Dickson's Lemma)**.** *Every monomial ideal is finitely generated.*

We next consider a special monomial ideal, associated with every polynomial ideal. From now on, we assume a fixed monomial ordering (e.g., graded reverse lexicographic), and define the notion of an initial ideal.

Definition 4.10 (Initial Ideal)**.** Denote by $\text{in}(f)$ the "largest" (or leading) monomial appearing in the polynomial $f \neq 0$. Consider an ideal $I \subset \mathbb{C}[x]$ with a fixed monomial ordering. The initial ideal of I, denoted by $\text{in}(I)$, is the monomial ideal generated by the leading monomials of all the elements in I, i.e.,

$$\text{in}(I) = \text{Ideal}[\{\text{in}(f) \mid f \in I \backslash \{0\}\}].$$

A monomial x^α is called **standard** if it does not belong to the initial ideal $\text{in}(I)$.

Given an ideal $I = \text{Ideal}[f_1, \dots, f_s]$, we can construct two monomial ideals associated with it:

- The initial ideal $\text{in}(I)$ as defined in Definition 4.10.
- The monomial ideal generated by the leading monomials of the generators, i.e.,
$$\text{Ideal}[\text{in}(f_1), \dots, \text{in}(f_s)].$$

Clearly, we have

$$\text{Ideal}[\text{in}(f_1), \dots, \text{in}(f_s)] \subseteq \text{in}(I).$$

Example 4.3 (Initial Ideal and the Ideal of Initials)**.** Consider the ideal

$$I = \text{Ideal}[x^3 - 1, x^2 + 1],$$

we have $1 \in I$ due to

$$1 = \frac{1}{2}(x-1)(x^3-1) - \frac{1}{2}(x^2 - x - 1)(x^2+1).$$

Therefore, $\text{in}(I) = I = \mathbb{C}[x]$. On the other hand, the monomial ideal generated by the leading monomials is

$$\text{Ideal}[x^3, x^2],$$

and clearly does not contain 1.

We have seen that in general these two monomial ideals are not the same. However, is it possible to find a set of generators for which the two monomial ideals are the same? This is exactly the notion of a Gröbner basis.

Definition 4.11 (Gröbner Basis)**.** Consider the polynomial ring $\mathbb{C}[x]$, a fixed monomial ordering, and an ideal I. A finite set of polynomials $\{g_1, \dots, g_s\} \subset I$ is a Gröbner basis of I if the initial ideal of I is generated by the leading terms of the g_i's, i.e.,

$$\text{in}(I) = \text{Ideal}[\text{in}(g_1), \dots, \text{in}(g_s)].$$

We have the result that every ideal has a Grobner basis.

Theorem 4.4 (Gröbner Basis)**.** *Every ideal I has a Grobner basis. Further,* $I = \text{Ideal}[g_1, \dots, g_s]$.

Theorem 4.4 indeed proves the Hilbert Basis Theorem 4.1.

Note that the Grobner basis as defined is not unique. It can be fixed to define a so-called **reduced Gröbner Basis**, which is unique when fixing a monomial ordering.

There exist numerical algorithms that can compute the Grobner basis. The most well-known algorithm is called Buchberger's algorithm, developed by Bruno Buchberger around 1965. The algorithm, however, is in general slow because it is double exponential time. Several software packages provide implementations that can compute the Grobner basis, for example Mathematica, Maple, Macaulay2, and Matlab.

Example 4.4 (Grobner Basis)**.** Consider the ideal $I = \text{Ideal}[x^2+y^2-2, x^2-y^2]$. The affine variety of I is finite and only contains four points $(\pm 1, \pm 1)$. It is easy to see that the given generators are not a Grobner basis.

Computing a Grobner basis (e.g., using the Matlab `gbasis` function) gives us

$$\{x^2 - 1, y^2 - 1\}.$$

The standard monomials are $\{1, x, y, xy\}$.

We will see that Grobner basis is a powerful tool with many applications in computational algebra. For a brief introduction to Grobner Basis, see (Sturmfels, 2005).

4.1.4 Quotient Ring

Given an ideal I, we can immediately define a notion of **equivalence classess**, where we identify two elements in the ring if and only if their difference is in the ideal.

For example, in the ring of integers $\mathbb{Z}$, the set of even integers is an ideal. We can identify two integers as the same if their difference is even. This effectively divides the ring $\mathbb{Z}$ into two equivalence classess, the set of odd and even integers.

We can do the same for the ring of polynomials $\mathbb{C}[x]$ given an ideal I.

Definition 4.12 (Congruence)**.** Let $I \subset \mathbb{C}[x]$ be an ideal. We are two polynomials $f, g \in \mathbb{C}[x]$ is **congruent** modulo I, written as

$$f \equiv g \mod I,$$

if $f - g \in I$.

It is easy to see that this is an equivalence relation, i.e., it is reflexive, symmetric, and transitive. Therefore, an ideal I partitions $\mathbb{C}[x]$ into equivalence classes, where two polynomials are the same if their difference belongs to the ideal. This allows us to define the notion of a quotient ring.

Definition 4.13 (Quotient Ring)**.** The quotient $\mathbb{C}[x]/I$ is the set of equivalence classes for congruence modulo I.

The quotient $\mathbb{C}[x]/I$ inherits the ring structure of $\mathbb{C}[x]$. With the addition and multiplication operations defined between equivalence classes, $\mathbb{C}[x]/I$ becomes a ring, known as the quotient ring. Indeed, given a polynomial p, the set of all polynomials equivalent to p is denoted as

$$[p] := \{p + q \mid q \in I\}.$$

We have $[p] = [q]$ if and only if $p - q \in I$. Given two equivalent classes $[p]$ and $[q]$, the addition and multiplication are defined as

$$[p] + [q] = [p + q], \quad [p] \cdot [q] = [pq].$$

One can verify that $(\mathbb{C}[x], +, \cdot)$ with $+$ and $\cdot$ defined above forms a ring structure.

The quotient ring $\mathbb{C}[x]/I$ is in fact a vector space, and the Grobner basis gives us a way to find a set of basis in the quotient ring! Let G be a Grobner basis of the ideal I, and recall from Definition 4.10 that a monomial is standard if it does not belong to the initial ideal $\text{in}(I)$. This implies that when the Grobner basis is available, the set of all **standard monomials** can be directly read off from the Grobner basis, as in Example 4.4. Let S be this set of standard monomials, then the basis of the quotient ring is simply

$$\{[s] \mid s \in S\}.$$

Note that the quotient ring needs not be finite-dimensional as the set of standard monomials can be infinite. Given such a basis of the quotient ring, we can associate every polynomial with a **normal form** that is a linear combination of the standard monomials.

Definition 4.14 (Normal Form)**.** Let G be a Grobner basis of the ideal $I \subset \mathbb{C}[x]$. Given any $p \in \mathbb{C}[x]$, there exists a unique polynomial $\bar{p}$, called the **normal form** of p such that

- p and $\bar{p}$ are congruent mod I, i.e., $p - \bar{p} \in I$.
- Only standard monomials appear in $\bar{p}$.

Note that since $p - \bar{p} \in I$, we have

$$p = \sum_i^s \lambda_i g_i + \bar{p}, \quad g_i \in G.$$

Therefore, the normal form $\bar{p}$ can be interpreted as the "reminder" of p divided by the ideal I. The unique property of the Grobner basis ensures that the normal form is unique and it is just a $\mathbb{C}$-combination of the standard monomials.

As a consequence, we can check the ideal membership of a given polynomial. Indeed, a polynomial p belongs to the ideal I if and only if its normal form is the zero polynomial.

See an example of using Grobner basis to compute the normal form in the Maple software.

4.1.5 Zero-dimensional Ideal

In practice, we are often interested in polynomial systems that have only a finite number of solutions, in which case the associated ideal is called zero-dimensional.

Definition 4.15 (Zero-Dimensional Ideal)**.** An ideal I is zero-dimensional if the associated affine variety $V_{\mathbb{C}}(I)$ is a finite set.

Note that the variety $V_{\mathbb{C}}(I)$ having a finite number of points is stronger than $V_{\mathbb{R}}(I)$ have a finite number of points.

The following result characterizes when an ideal is zero-dimensional.

Theorem 4.5 (Finiteness Theorem)**.** *Let $I \subset \mathbb{C}[x]$ be an ideal. Fix a monomial ordering and let G be a Grobner basis of I. Then the following statements are equivalent.*

1. *The ideal I is finite-dimensional (i.e., $V(I)$ is a finite set of points).*
2. *The quotient ring $\mathbb{C}[x]/I$ is a finite-dimensional vector space (i.e., there are a finite number of standard monomials).*
3. *For each $x_i, i = 1, \ldots, n$, there exists an integer $m_i \in \mathbb{N}$ such that $x_i^{m_i}$ is the leading monomial of some $g \in G$.*

Let us work out an example.

Example 4.5 (Zero-dimensional Ideal)**.** Consider the ideal in $\mathbb{C}[x, y]$:

$$I = \text{Ideal}[xy^3 - x^2, x^3y^2 - y].$$

Using the graded lexicographic monomial ordering, we obtain the reduced Grobner basis

$$G = \{x^3y^2 - y, x^4 - y^2, xy^3 - x^2, y^4 - xy\}.$$

We can see that x^4 is the leading monomial of $x^4 - y^2$, and y^4 is the leading monomial of $y^4 - xy$. Therefore, we know I is zero-dimensional. We can also see this by checking the number of standard monomials. The standard monomials

are those that do not belong to the initial ideal $\text{in}(I)$, which in the case of a Grobner basis, is simply

$$\text{in}(I) = \text{Ideal}[x^3y^2, x^4, xy^3, y^4].$$

This is a monomial ideal and hence is "closed upwards". The shaded gray areas in Fig. 4.1 shows all the monomials in the initial ideal, whose boundary forms a "staircase". The standard monomials, shown as blue squares in the figure, are those that are below the staircase:

$$1, x, x^2, x^3, y, xy, x^2y, x^3y, y^2, xy^2, x^2y^2, y^3.$$

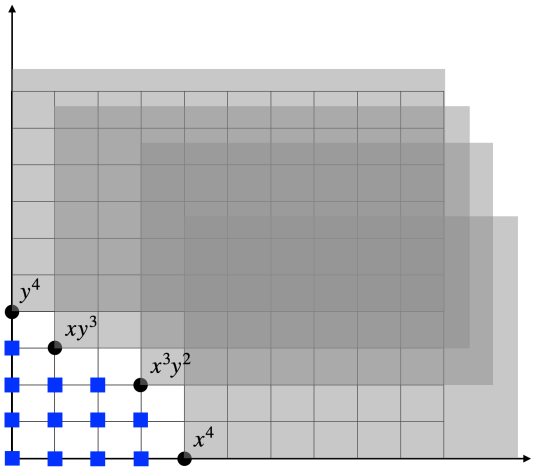

Figure 4.1: Initial ideal and the standard monomials with graded lexicographic monomial ordering.

We can change the monomial ordering and see what happens. Let us use the lexicographic monomial ordering this time. We obtain the reduced Grobner basis

$$G = \{x^2 - y^6, xy - y^4, y^{11} - y\}.$$

The monomial x^2 is the leading monomial of $x^2 - y^6$, and the monomial y^{11} is the leading monomial of $y^1 1 - y$. Hence, the ideal is verified to be zero-dimensional as well. A different monomial ordering does change the set of standard monomials. Fig. 4.2 shows the initial ideal and the standard monomials (blue squares) under the staircase:

$$1, x, y, y^2, y^3, y^4, y^5, y^6, y^7, y^8, y^9, y^{10}.$$

Though the set of standard monomials has changed, the number of standard monomials remains to be 12. This makes sense because the dimension of the quotient ring $\mathbb{C}[x]/I$ should not change.

The code for computing the Grobner basis can be found here.

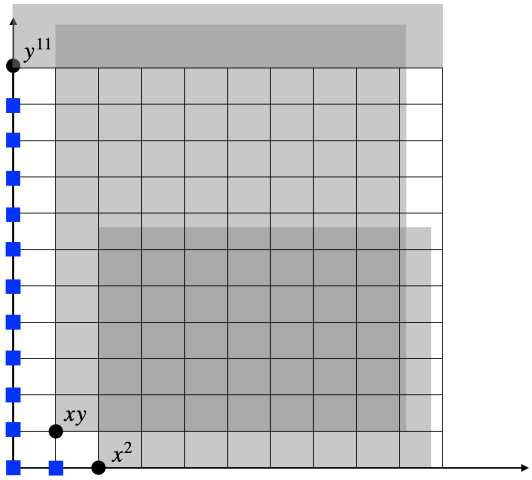

Figure 4.2: Initial ideal and the standard monomials with lexicographic monomial ordering.

If an ideal I is zero-dimensional, can we find all the points in the affine variety $V(I)$? The next result shows that the number of points in $V(I)$ (counting multiplicity) is precisely the number of standard monomials.

Theorem 4.6 (Number of Roots)**.** *If an ideal $I \subset \mathbb{C}[x]$ is zero-dimensional, then the number of standard monomials equals to the cardinality of $V(I)$, counting multiplicities.*

Once a set of standard monomials is found, we can also design a numerical procedure to compute the roots of the polynomial system of equations. For each $x_i, \ldots, x_n$, define the multiplication mapping

$$\mathcal{M}_{x_i} : \mathbb{C}[x]/I \to \mathbb{C}[x]/I, \quad [p] \to [x_i p].$$

It can be shown that $\mathcal{M}_{x_i}$ is a linear mapping. Let M_{x_i} be the matrix representation of this linear map using S, the set of standard monomials, as the basis of the quotient ring $\mathbb{C}[x]/I$. M_{x_i} is called the **companion matrix** or **multiplication matrix** of the ideal I with respect to x_i. The companion matrices $M_{x_1}, \ldots, M_{x_n}$ commute with each other and share common eigenvectors. The affine variety $V_{\mathbb{C}}(I)$ can be determined by eigenvalues of the companion matrices.

Theorem 4.7 (Stickelberger's Theorem)**.** *Let $I \subset \mathbb{C}[x]$ be a zero-dimensional ideal and let $M_{x_1}, \ldots, M_{x_n}$ be the companion matrices, then*

$$V_{\mathbb{C}}(I) = \{(\lambda_1, \ldots, \lambda_n) \mid \exists v \in \mathbb{C}^D \backslash \{0\}, M_{x_i} v = \lambda_i v, i = 1, \ldots, n\},$$

where D is the dimension of the quotient ring $\mathbb{C}[x]/I$.

With Stickelberger's Theorem, one can further show that a zero-dimensional ideal I is radical if and only if the companion matrices $M_{x_1}, \ldots, M_{x_n}$ are simultaneously diagonalizable, which occurs if and only if the cardinality $|V_{\mathbb{C}}(I)| = D$.

Let us make Stickelberger's Theorem concrete through an example.

Example 4.6 (Solving Polynomial Equations)**.** Consider the ideal $I \subset \mathbb{C}[x, y, z]$

$$I = \text{Ideal}[xy - z, yz - x, zx - y].$$

Choosing lexicographic monomial ordering, we get the Groebner basis

$$G = \{z - yx, y^2 - x^2, yx^2 - y, x^3 - x\}.$$

We can see that this ideal is zero-dimensional (due to z, y^2, x^3 being the leading monomials in G). The standard monomials are

$$S = \{1, x, x^2, y, yx\}.$$

We now need to form the matrices M_x, M_y, M_z. This can be done as follows.

$$\text{NormalForm}\left(x \cdot \begin{bmatrix} 1 \\ x \\ x^2 \\ y \\ yx \end{bmatrix}\right) = \begin{bmatrix} x \\ x^2 \\ x \\ xy \\ y \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}}_{M_x} \begin{bmatrix} 1 \\ x \\ x^2 \\ y \\ yx \end{bmatrix}$$

$$\text{NormalForm}\left(y \cdot \begin{bmatrix} 1 \\ x \\ x^2 \\ y \\ yx \end{bmatrix}\right) = \begin{bmatrix} y \\ xy \\ y \\ x^2 \\ x \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}}_{M_y} \begin{bmatrix} 1 \\ x \\ x^2 \\ y \\ yx \end{bmatrix}$$

$$\text{NormalForm}\left(z \cdot \begin{bmatrix} 1 \\ x \\ x^2 \\ y \\ yx \end{bmatrix}\right) = \begin{bmatrix} yx \\ y \\ xy \\ x \\ x^2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}}_{M_z} \begin{bmatrix} 1 \\ x \\ x^2 \\ y \\ yx \end{bmatrix}$$

We can then simultaneously diagonalize M_x, M_y, M_z using the following matrix

$$V = \frac{1}{4}\begin{bmatrix} 4 & 0 & -4 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & -1 & -1 \\ 0 & -1 & 1 & 1 & -1 \\ 0 & -1 & 1 & -1 & 1 \end{bmatrix}, \quad V^{-1} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & -1 & -1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & -1 & 1 & -1 \\ 0 & 1 & -1 & -1 & 1 \end{bmatrix}.$$

The result is

$$\begin{aligned} VM_xV^{-1} &= \text{diag}(0,1,1,-1,-1) \\ VM_yV^{-1} &= \text{diag}(0,1,-1,1,-1). \\ VM_zV^{-1} &= \text{diag}(0,1,-1,-1,1) \end{aligned} \tag{4.1}$$

Therefore, the five roots of the polynomial system is

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \left\{ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \\ 1 \end{pmatrix} \right\}.$$

In practice, instead of simultaneously diagonalizing the companion matrices, a Schur decomposition type method is used to compute the roots (Corless et al., 1997). We will see that in the homework.

4.1.6 Algebraic and Semialgebraic Sets

An **algebraic set** in $\mathbb{R}^n$ is the set of common **real** roots of a set of polynomials. For instance, the unit sphere is an algebraic set because it is real zero set of the polynomial $x_1^2 + \cdots + x_n^2 - 1$.

Intersections and unions of finitely many algebraic sets are again algebraic sets. A nonempty algebraic set is said to be **irreducible** if it cannot be written as a union of two distinct proper algebraic subsets; otherwise it is called reducible. Every algebraic set is the union of finitely many irreducible ones.

More general than algebraic sets are **semialgebraic sets**.

Definition 4.16 (Basic Semialgebraic Set)**.** A set $S \subset \mathbb{R}^n$ defined as

$$S = \{x \in \mathbb{R}^n \mid f_i(x) \rhd_i 0, i = 1, \ldots, \ell\},$$

where for each i, $\rhd_i$ is one of $\{\geq, >, =, \neq\}$ and $f_i(x) \in \mathbb{R}[x]$, is called a basic semialgebraic set.

A basic closed semialgebraic set is a set of the form

$$S = \{x \in \mathbb{R}^n \mid f_1 \geq 0, \ldots f_\ell(x) \geq 0\}.$$

Every basic semialgebraic set can be expressed with polynomial inequalities of the form $f(x) \geq 0$ and a single inequality $g \neq 0$.

Definition 4.17 (Semialgebraic Set)**.** A finite union of basic semialgebraic sets in $\mathbb{R}^n$ is called a semialgebraic set, and a finite union of basic closed semialgebraic sets is a closed semialgebraic set.

Semialgebraic sets are closed under finite unions, finite intersections, and complementation. The following theorem states that they are also closed under projections.

Theorem 4.8 (Tarski-Seidenberg Theorem)**.** *Let $S \subset \mathbb{R}^{k+n}$ be a semialgebraic set and $\pi : \mathbb{R}^{k+n} \to \mathbb{R}^n$ be the projection map that sends $(y, x) \mapsto x$. Then $\pi(S)$ is a semialgebraic set in $\mathbb{R}^n$.*

Note that an algebraic set is not closed under projection. For example consider the algebraic set defined by $xy = 1$. The projection of this algebraic set to x is defined by $x \neq 0$, which is not an algebraic set, but a semialgebraic set.

Semialgebraic functions are similarly defined. Let $f : D \to \mathbb{R}$ be a real-valued function where the domain D is a semialgebraic set. Then f is called a **semialgebraic function** if the graph $\{(x, y) \mid x \in D, f(x) = y\}$ is a semialgebraic set.

4.2 SOS and Nonnegative Polynomials

4.2.1 Nonnegative polynomials

Consider polynomials in n variables with real coefficients, i.e., the set $\mathbb{R}[x]$. A polynomial $p(x_1, \dots, x_n)$ is nonnegative if

$$p(x_1, \dots, x_n) \geq 0, \quad \forall (x_1, \dots, x_n) \in \mathbb{R}^n.$$

Of couse, a natural question is, given a polynomial $p(x)$, is it possible to efficiently decide whether $p(x)$ is nonnegative?

Univariate Polynomials. Let us start the discussion on univariate polynomials, i.e., polynomials in a single variable x. A univariate polynomial of degree d can be written as

$$p(x) = p_d x^d + p_{d-1} x^{d-1} + \cdots + p_1 x + p_0. \tag{4.2}$$

Without loss of generality, we can assume $p_d = 1$, in which case the polynomial is said to be **monic**. The univariate polynomial can also be written as

$$p(x) = p_d \prod_{i=1}^{d} (x - x_i), \tag{4.3}$$

with $x_i, i = 1, \dots, d$ its complex roots that may have multiplicities.

How do we decide if $p(x)$ is nonnegative? A simple necessary condition is that d must be even. Otherwise if d is odd, then either pushing $x \to \infty$ or $x \to -infinity$ will lead to $p(x)$ negative.

In certain cases it is easy to derive an explict characterization of nonnegativity.

Example 4.7 (Univaraite Quadratic Polynomial)**.** Let $p(x) = x^2 + p_1 x + p_0$ be a monic quadratic polynomial. Clearly, $p(x) \geq 0$ if and only if

$$\min_x p(x) \geq 0.$$

Since $p(x)$ is convex, its global minimum can be easily computed by setting its gradient to zero, which gives

$$x_\star = -\frac{p_1}{2}, \quad p(x_\star) = p_0 - \frac{p_1^2}{4}.$$

Therefore, $p(x)$ is nonnegative if and only if

$$4p_0 - p_1^2 \geq 0.$$

For general univariate polynomials, we describe a technique known as the Hermite method for deciding nonnegativity. Consider a monic polynomial as in (4.2) and (4.3) and define its associated **Hermite matrix** as the following $d \times d$ symmetric Hankel matrix

$$H_1(p) = \begin{bmatrix} s_0 & s_1 & \cdots & s_{d-1} \\ s_1 & s_2 & \cdots & s_d \\ \vdots & \vdots & \ddots & \vdots \\ s_{d-1} & s_d & \cdots & s_{2d-2} \end{bmatrix}, \quad s_k = \sum_{j=1}^{d} x_j^k,$$

where recall $x_j, j = 1, \ldots, d$ are the roots of p. The quantities s_k are known as the power sums. The follow lemma states that the power sums can be computed without computing the roots.

Lemma 4.1 (Newton Identities)**.** *The power sums s_k satisfy the following recursive equations known as the Newton identities:*

$$s_0 = d, \quad s_k = -1 \left(k p_{d-k} + \sum_{i=1}^{k-1} p_{d-i} s_{k-i} \right), k = 1, 2, \ldots.$$

After forming the Hermite matrix, its rank and signature tells us the number of complex and real roots.

Theorem 4.9 (Hermite Matrix)**.** *The rank of the Hermite matrix $H_1(p)$ is equal to the number of distinct complex roots. The signature of $H_1(p)$ is equal to the number of distinct real roots.*

Recall that given a symmetric matrix A, its **inertia**, denoted $\mathcal{I}(A)$, is the triplet (n_+, n_0, n_-), where n_+, n_0, n_- are the number of positive, zero, and negative eigenvalues, respectively. The **signature** of A is equal to $n_+ - n_-$.

With the Hermite matrix, we can decide nonnegativity.

Theorem 4.10 (Nonnegativity from Hermite Matrix)**.** *Let $p(x)$ be a monic polynomial of degree $2d$. Then the following statements are equivalent.*

1. *The polynomial $p(x)$ is strictly positive.*
2. *The polynomial $p(x)$ has no real roots.*
3. *The inertia of the Hermite matrix is $\mathcal{I}(H_1(p)) = (k, 2d-k, k)$ for some $1 \leq k \leq d$.*

We can use this result on the quadratic example before.

Example 4.8 (Hermite Matrix of A Quadratic Polynomial)**.** Consider the quadratic polynomial $p(x) = x^2 + p_1 x + p_0$. Using the Newton identifies, we have

$$s_0 = 2, \quad s_1 = -p_1, \quad s_2 = p_1^2 - 2p_0.$$

The Hermite matrix is then

$$H_1(p) = \begin{bmatrix} 2 & -p_1 \\ -p_1 & p_1^2 - 2p_0 \end{bmatrix}.$$

Let $\Delta = \det H_1(p) = p_1^2 - 4p_0$. The inertia of the Hermite matrix is

$$\mathcal{I}(H_1(p)) = \begin{cases} (0,0,2) & \text{if } \Delta > 0 \\ (0,1,1) & \text{if } \Delta = 0 \\ (1,0,1) & \text{if } \Delta < 0. \end{cases}$$

Thus, p is strictly positive if and only if $\Delta < 0$.

Multivariate Polynomials. Let us denote by $P_{n,2d}$ the set of nonnegative polynomials in n variables with degree up to $2d$, i.e.,

$$P_{n,2d} = \{f \in \mathbb{R}[x]_{2d} \mid f(x) \geq 0, \forall x \in \mathbb{R}^n\}.$$

There are

$$s(n, 2d) = \binom{n+2d}{2d}$$

coefficients for $f \in P_{n,2d}$. Notice that the constraint $f(x) \geq 0$ when fixing x is an affine constraint in the coefficients of p. Therefore, $P_{n,2d}$ is in fact a convex set. Further, it is a proper cone.

Theorem 4.11 (Cone of Nonnegative Polynomials)**.** *The set of nonnegative polynomials $P_{n,2d}$ is a proper cone (i.e., closed, convex, pointed, and solid) in $\mathbb{R}[x]_{2d} \sim \mathbb{R}^{s(n,2d)}$.*

Let us see an example of quadratic polynomials.

Example 4.9 (Cone of Nonnegative Quadratic Polynomials)**.** Consider the cone $P_{n,2}$, nonnegative polynomials in n variables of degree up to 2. Such polynomials can be written as

$$p(x) = x^\top Ax + 2b^\top x + c,$$

for some $A \in \mathbb{S}^n$, $b \in \mathbb{R}^n$, $c \in \mathbb{R}$. Observe that we can write

$$p(x) = \begin{bmatrix} x \\ 1 \end{bmatrix}^\top \begin{bmatrix} A & b \\ b^\top & c \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix},$$

therefore $p(x) \geq 0$ if and only if

$$\begin{bmatrix} A & b \\ b^\top & c \end{bmatrix} \succeq 0.$$

Thus, in this case the set $P_{n,2}$ is isomorphic to the positive semidefinite cone $\mathbb{S}_+^{n+1}$.

One may wonder is it the case that $P_{n,2d}$, being a convex cone, will always have nice descriptions as in the previous example?

Unfortunately the answer is no. In general the geometry of $P_{n,2d}$ is extremely complicated. In Example 4.9 we showed $P_{n,2}$ is isomorphic to $\mathbb{S}_+^{n+1}$ and hence it is **basic semialgebraic** (recall that the PSD cone can be described by a finite number of polynomial constraints). However, in general $P_{n,2d}$ is semialgebraic but not basic semialgebraic.

Example 4.10 (Semialgebraic Set of Nonnegative Polynomials)**.** Consider the quartic univariate polynomial

$$p(x) = x^4 + 2ax^2 + b, \quad a, b \in \mathbb{R}.$$

We are interested in finding conditions on a, b such that $p(x) \geq 0$ for any $x \in \mathbb{R}$.

A formal analysis can be done via the discriminant of $p(x)$. But here we will leverage a powerful tool known as **quantifier elimination**, using cylindrical algebraic decomposition, to directly give us the answer. For example, we can use the quantifier elimination function in the Maple software.

Fig. 4.3 shows my query to Maple. The quantifier elimination algorithm produces me a set of conditions on a, b such that $p(x) \geq 0$.

Plotting the conditions we get the blue region in Fig. 4.4. As we can see, the set of (a, b) such that $p(x)$ is nonnegative is indeed a convex set, but it is not a basic semialgebraic set in the sense that it cannot be written as the feasible set of a finite number of polynomial constraints. However, it is a semialgebraic set that can be described by the union of several basic semialgebraic sets.

Another issue with the convex set shown in Fig. 4.4 is that there exists a zero-dimensional face (a vertex) that is not exposed, which is the point $(0, 0)$. A non-exposed face is a known obstruction for a convex set to be the feasible set of a semidefinite program (Ramana and Goldman, 1995).

```
with(RegularChains) :
with(SemiAlgebraicSetTools) :
f := `&A`([x]), x^4 + 2·a·x^2 + b ≥ 0
        f := &A([x]), 0 ≤ x^4 + 2 a x^2 + b                    (1)
out := QuantifierElimination(f)
out := &or((a < 0) &and (a^2 − b < 0), (a                      (2)
   < 0) &and (a^2 − b = 0), (a = 0) &and (b
   = 0), (a = 0) &and (0 < b), (0 < a) &and
   (b = 0), (0 < a) &and (a^2 − b < 0), (0
   < a) &and (a^2 − b = 0), &and(0 < a, 0
   < b, 0 < a^2 − b))
```

Figure 4.3: Quantifier elimination in Maple.

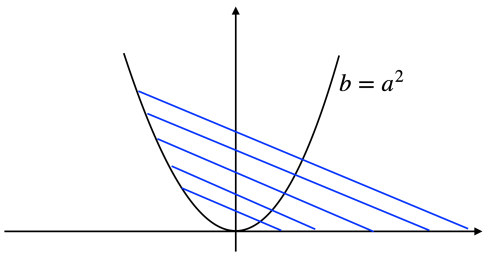

Figure 4.4: Set of coefficients such that the polynomial is nonnegative.

This shows the difficulty of working with nonnegative polynomials.

From the computational complexity perspective, the difficulty of working with nonnegative polynomials is seen by the fact that deciding polynomial nonnegativity for $2d \geq 4$ is known to be NP-hard in the worst case.

However, we do know very well a special class of nonnegative polynomials, the nonnegative polynomials that can be written as a sum of squares.

4.2.2 Sums-of-Squares Polynomials

A multivariate polynomial $p(x)$ is a **sum of squares** (SOS) if it can be written as the sum of squares of some other polynomials.

Definition 4.18 (Sum of Squares Polynomial)**.** A polynomial $p(x) \in \mathbb{R}[x]_{2d}$ is a sum of squares if there exist $q_1, \ldots, q_m \in \mathbb{R}[x]_d$ such that

$$p(x) = \sum_{k=1}^{m} q_k^2(x).$$

We will use $\Sigma_{n,2d}$ to denote the set of SOS polynomials in n variables of degree up to $2d$. Clearly, we have

$$\Sigma_{n,2d} \subseteq P_{n,2d},$$

because any SOS polynomial is necessarily nonnegative.

Similar to $P_{n,2d}$, the set of SOS polynomials is also a convex proper cone.

Theorem 4.12 (Cone of SOS Polynomials)**.** *The set of SOS polynomials $\Sigma_{n,2d}$ is a proper cone (i.e., closed, convex, pointed, and solid) in $\mathbb{R}[x]_{2d}$.*

One of the central questions in convex algebraic geometry is to understand the relationships between $P_{n,2d}$ and $\Sigma_{n,2d}$.

The first question is when is nonnegativity equal to sum of squares? David Hilbert showed that equality between $\Sigma_{n,2d}$ and $P_{n,2d}$ happens only in the following three cases:

- Univariate polynomials, $n = 1$
- Quadratic polynomials, $2d = 2$
- Bivariate quartics, $n = 2, 2d = 4$.

For all other cases, there always exist nonnegative polynomials that are not SOS. One of the most famous examples is the **Motzkin's polynomial** written as

$$M(x, y) = x^4y^2 + x^2y^4 + 1 - 3x^2y^2.$$

One can show that this polynomial is nonnegative but not SOS. Other examples of nonnegative but not SOS polynomials can be found in (Reznick, 2000).

Let us show that univariate nonnegative polynomials can always be written as a sum of squares.

Theorem 4.13 (Univaraite Nonnegative Polynomials are SOS)**.** *A univariate polynomial is nonnegative if and only if it is a sum of squares.*

Proof. Since $p(x)$ is univariate, we can factorize it as

$$p(x) = p_n \prod_j (x - r_j)^{n_j} \prod_k (x - a_k + ib_k)^{m_k} (x - a_k - ib_k)^{m_k}, \tag{4.4}$$

where r_j's are the real roots and $a_k \pm ib_k$'s are the complex roots. Because $p(x)$ is nonnegative, then $p_n > 0$ and the multiplicities of the real roots must be even, i.e., $n_j = 2s_j$.

Also note that

$$(x - a + ib)(x - a - ib) = (x - a)^2 + b^2.$$

Consequently we can write (4.4) as

$$p(x) = p_n \prod_j (x - r_j)^{2s_j} \prod_k ((x - a_k)^2 + b_k^2)^{m_k}.$$

Since products of sums of squares are still sums of squares and all the factors in the above expression are SOS, it follows that $p(x)$ is SOS.

Furthermore, the two-squares identify

$$(\alpha^2 + \beta^2)(\gamma^2 + \delta^2) = (\alpha\gamma - \beta\delta)^2 + (\alpha\delta + \beta\delta)^2$$

allows us to combine very partial product as a sum of only two squares:

$$p(x) = q_1^2(x) + q_2^2(x).$$

Therefore, it suffices to write every nonnegative univariate polynomial as the sum of only two squares. □

In the next section, we focus on how to numerically compute SOS decompositions.

4.3 Compute SOS Decompositions

Given a polynomial $p(x)$, how do we decide if $p(x)$ is SOS? We knew that deciding if $p(x)$ is nonnegative is generally NP-hard. The nice thing about SOS polynomials is that the decision problem is tractable and it is a convex semidefinite program.

4.3.1 Univaraite Polynomials

Consider a univariate polynomial $p(x)$ of degree $2d$

$$p(x) = p_{2d}x^{2d} + p_{2d-1}x^{2d-1} + \cdots + p_1 x + p_0.$$

Assume $p(x)$ is SOS and can be written as

$$p(x) = q_1^2(x) + \cdots + q_m^2(x). \tag{4.5}$$

Note that the degree of the polynomials q_k must be at most d since the leading term of $q_k^2(x)$ must have positive coefficients and there cannot be any cancellation in the highest power of x. Then, we can write

$$\begin{bmatrix} q_1(x) \\ q_2(x) \\ \vdots \\ q_m(x) \end{bmatrix} = V \begin{bmatrix} 1 \\ x \\ \vdots \\ x^d \end{bmatrix} = V[x]_d, \tag{4.6}$$

where recall $[x]_d$ is the vector of monomials in x of degree up to d. Plugging (4.6) into (4.5), we get

$$p(x) = \sum_{k=1}^{m} q_k^2(x) = (V[x]_d)^\top (V[x]_d) = [x]_d^\top V^\top V[x]_d = [x]_d^\top Q[x]_d. \tag{4.7}$$

This suggests the following characterization of univariate SOS polynomials.

Lemma 4.2 (Characterization of Univaraite SOS Polynomials)**.** *A univariate polynomial $p(x) \in \mathbb{R}[x]_{2d}$ is SOS if and only if there exists a symmetric matrix $Q \in \mathbb{S}^{d+1}$ that satisfies*

$$p(x) = [x]_d^\top Q[x]_d, \quad Q \succeq 0.$$

The matrix Q is typically called the **Gram matrix** of the SOS representation. Lemma 4.2 can be easily verified: one direction is clear from (4.7) and the other direction can be shown by factorizing a positive semidefinite matrix $Q = V^\top V$.

Although it may not be immediately clear, but the nice property of Lemma 4.2 is that it says deciding if a univariate polynomial is SOS (and finding its SOS decomposition) is in fact a convex semidefinite program!

To see this, note that the constraint $Q \succeq 0$ is a convex PSD constraint and the constraint $p(x) = [x]_d^\top Q[x]_d$ leads to a finite number of affine constraints on the entries of Q by "matching coefficients". In particular, let us index the rows and columns of Q from 0 to d, and expand

$$[x]_d^\top Q[x]_d = \sum_{k=0}^{2d} \left(\sum_{i+j=k} Q_{ij} \right) x^k.$$

It now becomes clear that $p(x) = [x]_d^\top Q[x]_d$ simply askes

$$\sum_{i+j=k} Q_{ij} = p_k, \quad k = 0, \dots, 2d.$$

It is easier to work this out via an example.

Example 4.11 (SOS Decomposition of Univaraite Polynomials)**.** Consider the univariate polynomial

$$p(x) = x^4 + 4x^3 + 6x^2 + 4x + 5.$$

To decide if this polynomial is SOS, we write the polynomial equality constraint

$$\begin{aligned} p(x) &= \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^\top \begin{bmatrix} Q_{00} & Q_{01} & Q_{02} \\ Q_{01} & Q_{11} & Q_{12} \\ Q_{02} & Q_{12} & Q_{22} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} \\ &= Q_{22}x^4 + 2Q_{12}x^3 + (Q_{11} + 2Q_{02})x^2 + 2Q_{01}x + Q_{00}. \end{aligned} \tag{4.8}$$

Matching coefficients, we get the constraints

$$\begin{aligned} x^4 : &\quad Q_{22} = 1 \\ x^3 : &\quad 2Q_{12} = 4 \\ x^2 : &\quad Q_{11} + 2Q_{02} = 6 \\ x : &\quad 2Q_{01} = 4 \\ 1 : &\quad Q_{00} = 5 \end{aligned} \tag{4.9}$$

We need to find a $Q \succeq 0$ that satisfies the above constraints, which is clearly a convex SDP. In this case, the SDP is feasible and we can find a solution

$$Q = \begin{bmatrix} 5 & 2 & 0 \\ 2 & 6 & 2 \\ 0 & 2 & 1 \end{bmatrix}.$$

Factorizing $Q = V^\top V$ with

$$V = \begin{bmatrix} 0 & 2 & 1 \\ \sqrt{2} & \sqrt{2} & 0 \\ \sqrt{3} & 0 & 0 \end{bmatrix},$$

we obtain the SOS decomposition

$$p(x) = (x^2 + 2x)^2 + 2(1 + x)^2 + 3,$$

4.3.2 Multivariate Polynomials

Deciding if a multivariate polynomial is SOS is almost identical to what we have shown for univariate polynomials. Consider a multivariate polynomial $p(x)$ in n variables of degree $2d$:

$$p(x) = \sum_{\alpha} p_\alpha x^\alpha, \quad \alpha \in \mathcal{F}_{n,2d} \coloneqq \{(\alpha_1, \dots, \alpha_n) \in \mathbb{N}^n \mid \alpha_1 + \cdots + \alpha_n \leq 2d\}.$$

Let

$$[x]_d \coloneqq \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \\ x_1^2 \\ x_1 x_2 \\ \vdots \\ x_n^d \end{bmatrix} = [x^\alpha]_{\alpha \in \mathcal{F}_{n,d}} \tag{4.10}$$

be the standard basis of monomials with degree up to d. Then in parallel to Lemma 4.2, we have that $p(x)$ is SOS if and only if there exists a positive semidefinite Gram matrix.

Lemma 4.3 (Characterization of Multivariate SOS Polynomials)**.** *A multivariate polynomial $p(x) \in \mathbb{R}[x]_{2d}$ is SOS if and only if there exists a symmetric matrix $Q \in \mathbb{S}^{s(n,d)}$ that satisfies*

$$p(x) = [x]_d^\top Q [x]_d, \quad Q \succeq 0.$$

Let us index the monomials in the basis (4.10), as well as the rows and columns of Q, using their exponents. Then by matching coefficients of the polynomial equation $p(x) = [x]_d^\top Q [x]_d$, Lemma 4.3 is equivalent to finding $Q \in \mathbb{S}^{s(n,d)}$ that satisfies

$$p_\alpha = \sum_{\beta+\gamma=\alpha} Q_{\beta\gamma}, \forall \alpha \in \mathcal{F}_{n,2d}, \quad Q \succeq 0. \tag{4.11}$$

That is, for every $\alpha \in \mathcal{F}_{n,2d}$, we search for all possible pairs of $\beta, \gamma \in \mathcal{F}_{n,d}$ such that $\beta + \gamma = \alpha$, the sum of $Q_{\beta\gamma}$ should be equal to p_α by the virtue of matching coefficients. **Note that** (4.11) **boils down to solving an SDP with matrix size** $s(n,d) \times s(n,d)$ **and** $s(n,2d)$ **affine constraints.** Therefore, it grows quickly with n and d.

Let us work out a simple example.

Example 4.12 (SOS Decomposition of a Multivariate Polynomial)**.** We want to check if the following polynomial in two variables is SOS:

$$p(x) = 2x_1^4 + 5x_2^4 - x_1^2 x_2^2 + 2x_1^3 x_2 + 2x_1 + 2.$$

Since $\deg(p) = 4$, we pick the standard monomial basis of degree up to 2

$$[x]_2 = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix},$$

and write down the polynomial equality constraint

$$p(x) = [x]_2^\top Q [x]_2 = [x]_2^\top \begin{bmatrix} Q_{00,00} & Q_{00,10} & Q_{00,01} & Q_{00,20} & Q_{00,11} & Q_{00,02} \\ * & Q_{10,10} & Q_{10,01} & Q_{10,20} & Q_{10,11} & Q_{10,02} \\ * & * & Q_{01,01} & Q_{01,20} & Q_{01,11} & Q_{01,02} \\ * & * & * & Q_{20,20} & Q_{20,11} & Q_{20,02} \\ * & * & * & * & Q_{11,11} & Q_{11,02} \\ * & * & * & * & * & Q_{02,02} \end{bmatrix} [x]_2$$

By matching coefficients, we obtain

$$s(n, 2d) = s(2, 4) = \binom{2+4}{4} = 15$$

affine constraints on the entries of Q, which has size 6×6.

Solving the SDP, we obtain a solution

$$Q = \frac{1}{3} \begin{bmatrix} 6 & 3 & 0 & -2 & 0 & -2 \\ 3 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \\ -2 & 0 & 0 & 6 & 3 & -4 \\ 0 & 0 & 0 & 3 & 5 & 0 \\ -2 & 0 & 0 & -4 & 0 & 15 \end{bmatrix}.$$

Choice of Basis. It is worth noting that so far, when computing SOS decompositions, we have been using the standard monomial basis $[x]_d$. This is the most popular choice in practice, but it is not necessarily the only choice. See Chapter 3.1.5 of (Blekherman et al., 2012) for other basis choices.

4.4 SOS Programming

We have seen that finding SOS decompositions of given polynomials can be written as semidefinite programs. It is clear to see that we can do more than just finding SOS decompositions – we can also formulate optimization problems subject to SOS constraints, known as SOS programming.

Definition 4.19 (SOS Program)**.** An SOS optimization problem or SOS program is a convex optimization problem of the form

$$\begin{array}{ll} \max\limits_{y} & b_1 y_1 + \cdots + b_m y_m \\ \text{s.t.} & p_i(x;y) \text{ is SOS in } \mathbb{R}[x], \quad i = 1, \ldots, k. \end{array} \tag{4.12}$$

where $b = (b_1, \ldots, b_m) \in \mathbb{R}^m$ is a given constant vector, $y = (y_1, \ldots, y_m)$ is the unknown variable to be optimized, and

$$p_i(x;y) = a_{i0}(x) + y_1 a_{i1}(x) + \cdots + y_m a_{im}(x), \quad i = 1, \ldots, k,$$

are polynomials in x with coefficientss affine in y, with $a_{ij}(x), i = 1, \ldots, k, j = 0, \ldots, m$ given polynomials.

For readers seeing SOS programs for the first time, it is critical to realize that **the variable x in an SOS program** (4.12) **is "dummy"**, in the sense that it is only used to formulate the problem, but not really being optimized. The variable $y \in \mathbb{R}^m$ is the optimization variable.

To see this point more clearly, we will show that the SOS program (4.12) can be reformulated as a convex semidefinite program (or in general a conic optimization problem). Let

$$d_i = \lceil \deg(p_i)/2 \rceil, \quad i = 1, \ldots, k,$$

where importantly, the degree of p_i is taken w. r. t. x (but not y). Then we can clearly see that the constraint "$p_i(x;y)$ is SOS in $\mathbb{R}[x]$" is equivalent to

$$p_i(x;y) = [x]_{d_i}^\top Q_i [x]_{d_i}, \quad Q_i \in \mathbb{S}_+^{s(n,d_i)}, \quad i = 1, \ldots, k, \tag{4.13}$$

where recall $[x]_{d_i}$ is the standard monomial basis in x of degree up to d_i. By matching coefficients in equation (4.13), we obtain a set of affine constraints in y and Q_i, while the variable x is simply removed! Therefore, the SOS program (4.12) is equivalent to a conic optimization in the variable

$$(y, Q_1, \ldots, Q_k) \in \mathbb{R}^m \times \mathbb{S}_+^{s(n,d_1)} \times \cdots \times \mathbb{S}_+^{s(n,d_k)}.$$

The number of affine constraints is

$$s(n, 2d_1) + \cdots + s(n, 2d_k).$$

Let us make this observation concrete using a simple example.

Example 4.13 (SOS Programming)**.** Consider the following SOS program

$$\begin{array}{ll} \max\limits_{y} & y_1 + y_2 \\ \text{s.t.} & x^4 + y_1 x + (2 + y_2) \text{ is SOS} \\ & (y_1 - y_2 + 1)x^2 + y_2 x + 1 \text{ is SOS} \end{array} \tag{4.14}$$

The first SOS constraint is equivalent to

$$x^4 + y_1 x + (2 + y_2) = [x]_2^\top Q_1 [x]_2 = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^\top \underbrace{\begin{bmatrix} Q_{1,00} & Q_{1,01} & Q_{1,02} \\ Q_{1,01} & Q_{1,11} & Q_{1,12} \\ Q_{1,02} & Q_{1,12} & Q_{1,22} \end{bmatrix}}_{Q_1} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}.$$

Matching coefficients, we obtain

$$\begin{aligned} x^4 : &\quad 1 = Q_{1,22} \\ x^3 : &\quad 0 = 2Q_{1,12} \\ x^2 : &\quad 0 = Q_{1,11} + 2Q_{1,02} \\ x : &\quad y_1 = 2Q_{1,01} \\ 1 : &\quad 2 + y_2 = Q_{1,00} \end{aligned} \tag{4.15}$$

The second SOS constraint is equivalent to

$$(y_1 - y_2 + 1)x^2 + y_2 x + 1 = [x]_1^\top Q_2 [x]_1 = \begin{bmatrix} 1 \\ x \end{bmatrix}^\top \underbrace{\begin{bmatrix} Q_{2,00} & Q_{2,01} \\ Q_{2,01} & Q_{2,11} \end{bmatrix}}_{Q_2} \begin{bmatrix} 1 \\ x \end{bmatrix}.$$

Matching coefficients, we obtain

$$\begin{aligned} x^2 : &\quad y_1 - y_2 + 1 = Q_{2,11} \\ x : &\quad y_2 = 2Q_{2,01} \\ 1 : &\quad 1 = Q_{2,00} \end{aligned} \tag{4.16}$$

Therefore, we obtain a conic optimization in

$$(y_1, y_2, Q_1, Q_2)$$

subject to the affine constraints in (4.15) and (4.16), plus $Q_1 \succeq 0, Q_2 \succeq 0$.

For this simple example, it is tractable to collect the linear constraints by hand. When the SOS program grows larger, we can use existing software packages to perform the conversion for us.

In the next, we show how to use the software package SOSTOOLS (Prajna et al., 2002) to implement SOS programming in Matlab.

I first create the dummy polynomial variable x and the decision variable y.

```
dim_y = 2;
dim_x = 1;
x = mpvar('x',1); % dummy x
prog = sosprogram(x);
```

```
y_name = {}; % decision variable y
for i = 1:dim_y
    y_name{i} = sprintf('y_%d',i);
end
y = dpvar(y_name);
y = y(:);
prog = sosdecvar(prog,y);
```

I then define the two polynomials in the SOS constraints, two SOS polynomials, and enforce them to the equal.

```
% two polynomials
p1 = x^4 + y(1)*x + (2 + y(2));
p2 = (y(1)-y(2)+1)*x^2 + y(2)*x + 1;
% two SOS polynomials
[prog, sig1] = sossosvar(prog,monomials(x,0:2));
[prog, sig2] = sossosvar(prog,monomials(x,0:1));
% matching coefficients
prog = soseq(prog,p1-sig1);
prog = soseq(prog,p2-sig2);
```

Finally, I set the objective function, choose a solver, and solve the SOS program.

```
% objective
prog = sossetobj(prog,-(y(1)+y(2)));
% choose solver
options.solver = 'mosek';
prog = sossolve(prog,options);
```

After solving the SOS program, I extract the optimal solution

```
% get solution
ystar = double(sosgetsol(prog,y));
```

which produces

$$y_\star = (6.6189, 3.8716).$$

How do I get the solutions to Q_1 and Q_2? If I go to the field `prog.solinfo.x`, I see a vector of dimension 15, which is $2 + 3^2 + 2^2$. Therefore, I can extract $Q_{1\star}$ and $Q_{2\star}$ with

```
Q1star = reshape(prog.solinfo.x(2+1:2+9),3,3);
Q2star = reshape(prog.solinfo.x(2+9+1:end),2,2);
```

which produces

$$Q_{1\star} = \begin{bmatrix} 5.8716 & 3.3095 & -1.3990 \\ 3.3095 & 2.7980 & 0 \\ -1.3990 & 0 & 1 \end{bmatrix}, \quad Q_{2\star} = \begin{bmatrix} 1 & 1.9358 \\ 1.9358 & 3.7473 \end{bmatrix}.$$

The implementation can be found here.

The implementation in SOSTOOLS is a bit unsatisfying because (i) one needs to separately define x and y, (ii) the extraction of Q_1 and Q_2 is not very intuitive. The following implementation shows how to solve the same SOS program using YALMIP (Lofberg, 2004).

```
% define all variables
x = sdpvar(1);
y = sdpvar(2,1);
Q1 = sdpvar(3,3);
Q2 = sdpvar(2,2);
% define polynomials
p1 = x^4 + y(1)*x + (2 + y(2));
p2 = (y(1)-y(2)+1)*x^2 + y(2)*x + 1;
% SOS polynomials
sig1 = monolist(x,2)' * Q1 * monolist(x,2);
sig2 = monolist(x,1)' * Q2 * monolist(x,1);
% define all constraints
F = [Q1>=0, Q2>=0,...
    coefficients(p1-sig1,x)==0,...
    coefficients(p2-sig2,x)==0];
% objective
obj = -(y(1)+y(2));
% solve
options = sdpsettings('solver','mosek');
optimize(F,obj,options);
% extract solution
ystar = value(y);
Q1star = value(Q1);
Q2star = value(Q2);
```

We get the same solution as using SOSTOOLS. The implementation in YALMIP can be found here.

4.5 Positivstellensatz

We have seen that in the univariate case, a polynomial is nonnegative if and only if it is SOS; while in the multivariate case, being SOS is a sufficient but not necessary condition for nonnegativity in general (e.g., think of the Motzkin's polynomial). Those characterizations are stated with respect to $\mathbb{R}^n$. What if we are interested in nonnegativity over only a subset of $\mathbb{R}^n$?

4.5.1 Univaraite Intervals

The first result we present here is w.r.t. univariate intervals.

Theorem 4.14 (Nonnegativity Over Univaraite Intervals)**.** *A univariate polynomial $p(x)$ is nonnegative on $[0, \infty]$ if and only if it can be written as*

$$p(x) = s(x) + x \cdot t(x) \tag{4.17}$$

where $s(x)$ and $t(x)$ are both SOS polynomials. If $\deg(p) = 2d$, then we have $\deg(s) \leq 2d$ and $\deg(t) \leq 2d - 2$, while if $\deg(p) = 2d + 1$, then $\deg(s) \leq 2d, \deg(t) \leq 2d$.

Similarly, a univariate polynomial $p(x)$ is nonnegative on the interval $[a, b]$ with $a < b$ if and only if it can be written as

$$\begin{cases} p(x) = s(x) + (x-a)(b-x) \cdot t(x) & \text{if } \deg(p) = 2d, \\ p(x) = (x-a) \cdot s(x) + (b-x) \cdot t(x) & \text{if } \deg(p) = 2d+1 \end{cases} \tag{4.18}$$

where both $s(x), t(x)$ are SOS polynomials. In the first case, $\deg(s) \leq 2d$, $\deg(t) \leq 2d - 2$. In the second case, $\deg(s) \leq 2d, \deg(t) \leq 2d$.

A simple example of the above theorem is that the polynomial $p(x) = x^3$ is nonnegative on the interval $[0, \infty]$ (but not on $\mathbb{R}$) because we can write $p(x) = x \cdot x^2$ where x^2 is SOS.

It should be noted that the "if" direction is clear, i.e., when $p(x)$ can be written as in (4.17) and (4.18), it certifies nonnegativity because the polynomials $x, x - a, b - x, (x - a)(b - x)$ are all nonnegative on the respective intervals. The "only if" direction is nontrivial and it states that it is sufficient to consider the decompositions in (4.17) and (4.18) with bounded degrees on $s(x)$ and $t(x)$. It should also be noted that finding $s(x)$ and $t(x)$ can be done using SOS programming as introduced above.

4.5.2 Affine Variety

Bibliography

Alizadeh, F., Haeberly, J.-P. A., and Overton, M. L. (1998). Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM Journal on Optimization*, 8(3):746–768. 45, 47

Bertsekas, D., Nedic, A., and Ozdaglar, A. (2003). *Convex analysis and optimization*, volume 1. Athena Scientific. 11

Bertsimas, D. and Tsitsiklis, J. N. (1997). *Introduction to linear optimization*, volume 6. Athena scientific Belmont, MA. 20, 22

Blekherman, G., Parrilo, P. A., and Thomas, R. R. (2012). *Semidefinite optimization and convex algebraic geometry*. SIAM. 32, 98

Bochnak, J., Coste, M., and Roy, M.-F. (2013). *Real algebraic geometry*, volume 36. Springer Science & Business Media. 73

Boyd, S., El Ghaoui, L., Feron, E., and Balakrishnan, V. (1994). *Linear matrix inequalities in system and control theory*. SIAM. 49

Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press. 11

Briales, J., Kneip, L., and Gonzalez-Jimenez, J. (2018). A certifiably globally optimal solution to the non-minimal relative pose problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 145–154. 69, 72

Corless, R. M., Gianni, P. M., and Trager, B. M. (1997). A reordered schur factorization method for zero-dimensional polynomial systems with multiple roots. In *Proceedings of the 1997 international symposium on Symbolic and algebraic computation*, pages 133–140. 87

Cox, D., Little, J., and OShea, D. (2013). *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Springer Science & Business Media. 73

Dummit, D. S. and Foote, R. M. (2004). *Abstract algebra*, volume 3. Wiley Hoboken. 73

Eriksson, A., Olsson, C., Kahl, F., and Chin, T.-J. (2018). Rotation averaging and strong duality. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 127–135. 64

Fazel, M., Ge, R., Kakade, S., and Mesbahi, M. (2018). Global convergence of policy gradient methods for the linear quadratic regulator. In *International conference on machine learning*, pages 1467–1476. PMLR. 52

Garcia-Salguero, M., Briales, J., and Gonzalez-Jimenez, J. (2021). Certifiable relative pose estimation. *Image and Vision Computing*, 109:104142. 59

Goemans, M. X. and Williamson, D. P. (1995). Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145. 62

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press. 71

Helmberg, C., Rendl, F., Vanderbei, R. J., and Wolkowicz, H. (1996). An interior-point method for semidefinite programming. *SIAM Journal on optimization*, 6(2):342–361. 49

Holmes, C. and Barfoot, T. D. (2023). An efficient global optimality certificate for landmark-based slam. *IEEE Robotics and Automation Letters*, 8(3):1539–1546. 59

Hu, B., Zhang, K., Li, N., Mesbahi, M., Fazel, M., and Başar, T. (2023). Toward a theoretical foundation of policy optimization for learning control policies. *Annual Review of Control, Robotics, and Autonomous Systems*, 6:123–158. 51

Kojima, M., Shindoh, S., and Hara, S. (1997). Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices. *SIAM Journal on Optimization*, 7(1):86–125. 49

Lang, S. (2012). *Algebra*, volume 211. Springer Science & Business Media. 73

Lofberg, J. (2004). Yalmip: A toolbox for modeling and optimization in matlab. In *2004 IEEE international conference on robotics and automation (IEEE Cat. No. 04CH37508)*, pages 284–289. IEEE. 102

Mansour, Y., Mohri, M., and Rostamizadeh, A. (2009). Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*. 55

Mohammadi, H., Zare, A., Soltanolkotabi, M., and Jovanović, M. R. (2021). Convergence and sample complexity of gradient methods for the model-free linear–quadratic regulator problem. *IEEE Transactions on Automatic Control*, 67(5):2435–2450. 51

Monteiro, R. D. (1997). Primal–dual path-following algorithms for semidefinite programming. *SIAM journal on Optimization*, 7(3):663–678. 48

Monteiro, R. D. (1998). Polynomial convergence of primal-dual algorithms for semidefinite programming based on the monteiro and zhang family of directions. *SIAM Journal on Optimization*, 8(3):797–812. 48

Nie, J. (2023). *Moment and Polynomial Optimization*. SIAM. 37, 39

Nistér, D. (2004). An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770. 69

Nocedal, J. and Wright, S. J. (1999). *Numerical optimization*. Springer. 42

Prajna, S., Papachristodoulou, A., and Parrilo, P. A. (2002). Introducing sostools: A general purpose sum of squares programming solver. In *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.*, volume 1, pages 741–746. IEEE. 100

Ramana, M. and Goldman, A. J. (1995). Some geometric results in semidefinite programming. *Journal of Global Optimization*, 7(1):33–50. 91

Ramana, M. V. (1997). An exact duality theory for semidefinite programming and its complexity implications. *Mathematical Programming*, 77:129–162. 35

Reznick, B. (2000). Some concrete aspects of hilbert's 17th problem. *Contemporary mathematics*, 253:251–272. 94

Rockafellar, R. T. (1970). *Convex Analysis*. Princeton university press. 11

Rosen, D. M., Carlone, L., Bandeira, A. S., and Leonard, J. J. (2019). Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. *The International Journal of Robotics Research*, 38(2-3):95–125. 64

Schacke, K. (2004). On the kronecker product. *Master's thesis, University of Waterloo*. 43

Shor, N. Z. (1987). Quadratic optimization problems. *Soviet Journal of Computer and Systems Sciences*, 25:1–11. 57

Sturmfels, B. (2005). What is... a grobner basis? *Notices-American Mathematical Society*, 52(10):1199. 81

Tang, Y., Lasserre, J.-B., and Yang, H. (2023). Uncertainty quantification of set-membership estimation in control and perception: Revisiting the minimum enclosing ellipsoid. *arXiv preprint arXiv:2311.15962*. 17

Todd, M. J. (2001). Semidefinite optimization. *Acta Numerica*, 10:515–560. 36

Todd, M. J., Toh, K.-C., and Tütüncü, R. H. (1998). On the nesterov–todd direction in semidefinite programming. *SIAM Journal on Optimization*, 8(3):769–796. 49

Wang, L. and Singer, A. (2013). Exact and stable recovery of rotations for robust synchronization. *Information and Inference: A Journal of the IMA*, 2(2):145–193. 64

Wang, Q., Zhou, X., Hariharan, B., and Snavely, N. (2020). Learning feature descriptors using camera pose supervision. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 757–774. Springer. 69

Wolkowicz, H., Saigal, R., and Vandenberghe, L. (2000). *Handbook of semidefinite programming: theory, algorithms, and applications*, volume 27. Springer. 30, 31, 61

Yang, H., Liang, L., Carlone, L., and Toh, K.-C. (2022). An inexact projected gradient method with rounding and lifting by nonlinear programming for solving rank-one semidefinite relaxation of polynomial optimization. *Mathematical Programming*, pages 1–64. 26

Zhang, Y. (1998). On extending some primal–dual interior-point algorithms from linear programming to semidefinite programming. *SIAM Journal on Optimization*, 8(2):365–386. 48