Lecture 11: Semidefinite programming and spectrahedra CSE 599I, Spring 2021
Instructor: James R. Lee

1 SDPs and spectrahedra

Let us denote by $\mathbb{S}_n$ the set of $n \times n$ real symmetric matrices equipped with the Frobenius inner product:

$$\langle A, B\rangle = \mathrm{Tr}(AB).$$

Consider the subset $\mathbb{S}_n^+ := \{X \in \mathbb{S}_n : X \succeq 0\}$ of PSD matrices. This is a closed convex cone, over which it is possible to minimize (sufficiently nice) convex functions efficiently. A *spectrahedron* is the intersection of $\mathbb{S}_n^+$ with an affine subspace of $\mathbb{S}_n$. The general form of a spectrahedron is

$$\mathcal{U} = \{X \in \mathbb{S}_n^+ : \langle X, A_1\rangle = b_1, \ldots, \langle X, A_m\rangle = b_m\},$$

for some $A_1, \ldots, A_m \in \mathbb{S}_n$, i.e., the set of PSD matrices satisfying a system of linear equations.

Note that spectrahedra are matrix versions of convex polyhedra, which can be seen (after a suitable affine transformation) as the sets $\mathbb{R}_+^n \cap \mathcal{L}$, where $\mathcal{L}$ is an affine subspace of $\mathbb{R}^n$.

Affine-equivalent formulations. An *affine transformation* $F : \mathbb{R}^d \to \mathbb{R}^d$ is a map of the form $F(x) = Ax + b$ where $A \in \mathbb{M}_d(\mathbb{R}), b \in \mathbb{R}^d$. An *affine subspace of* $\mathbb{R}^d$ is a set of the form $F(\mathbb{R}^d)$ for some affine transformation F. We will generally only be concerned with spectrahedra up to affine transformations, and thus the following two alternate ways of characterizing spectrahedra will be useful.

Slack variables. One could, more generally consider, convex sets of the form

$$\mathcal{U} = \{X \in \mathbb{S}_n^+ : \langle X, A_1\rangle \leqslant b_1, \ldots, \langle X, A_m\rangle \leqslant b_m\},$$

but there is a standard method of envisioning such a set as a spectrahedron in $\mathbb{S}_{n+m}$ using slack variables:

$$\tilde{\mathcal{U}} = \left\{X \oplus \mathrm{diag}(s_1, \ldots, s_m) \in \mathbb{S}_{n+m}^+ : \langle X, A_1\rangle = b_1 + s_1, \ldots, \langle X, A_m\rangle = b_m + s_m\right\}.$$

Here we use the notation: For $X \in \mathbb{M}_d, Y \in \mathbb{M}_{d'}, X \oplus Y \in \mathbb{M}_{d+d'}$ is the linear mapping defined by $(X \oplus Y)(u, v) = (Xu, Yv)$ for $(u, v) \in \mathbb{R}^d \oplus \mathbb{R}^{d'}$. In matrix form, $X \oplus Y$ is given by the matrix $\begin{bmatrix} X & 0 \\ 0 & Y \end{bmatrix}$.

Linear matrix inequality. Consider a spectrahedron $\mathcal{U} = \mathbb{S}_n^+ \cap \mathcal{L}$, where $\mathcal{L}$ is an affine subspace of $\mathbb{S}_n$ spanned by $A_0, A_1, \ldots, A_d \in \mathbb{S}_n^+$. Then $\mathcal{U}$ is affine-equivalent to the set

$$\tilde{\mathcal{U}} = \left\{y \in \mathbb{R}^d : A_0 + y_1 A_1 + \cdots + y_d A_d \succeq 0\right\}.$$

Thus a general spectrahedron is affine-equivalent to a set of the form

$$\mathcal{V} = \left\{y \in \mathbb{R}^d : y_1 A_1 + \cdots + y_d A_d \preceq B\right\}. \tag{1.1}$$

1.1 Semidefinite programs

Semidefinite programming (SDP) involves optimizing linear functions over spectrahedra: For some $C \in \mathbb{S}_n$ and a spectrahedron $\mathcal{U}$, this involves optimization problems of the form

$$\min \{\langle C, X\rangle : X \in \mathcal{U}\}.$$

Especially for relaxations of combinatorial optimization problems, a popular way of specifying an SDP is via a corresponding *vector program:*

$$\begin{aligned} \text{minimize} \quad & \sum_{i,j} c_{ij} \langle v_i, v_j \rangle \\ \text{subject to} \quad & v_1, v_2, \ldots, v_n \in \mathbb{R}^n, \\ & \sum_{i,j} a_{ij}^{(1)} \langle v_i, v_j \rangle \leqslant b_1, \\ & \ldots \\ & \sum_{i,j} a_{ij}^{(m)} \langle v_i, v_j \rangle \leqslant b_m. \end{aligned}$$

This can be written as an SDP using the equivalence between positive semidefinite matrices and the corresponding Gram matrices $X_{ij} = \langle v_i, v_j \rangle$. In general, such a program can be solved, up to accuracy ε, in time poly$(n, m, 1/\varepsilon)$.

MAX-CUT relaxation. As an example, the classic Goemans-Williamson relaxation of MAX-CUT on an n-vertex graph $G = (V, E)$ with nonnegative edge weights $\{w_{ij} : ij \in E\}$ is written as

$$\begin{aligned} \text{maximize} \quad & \frac{1}{2} \sum_{ij \in E} w_{ij} \|v_i - v_j\|^2 \\ \text{subject to} \quad & v_1, v_2, \ldots, v_n \in \mathbb{R}^n, \\ & \|v_i\|^2 = 1, \quad i = 1, \ldots, n, \end{aligned}$$

where $\|\cdot\|$ is the Euclidean norm on $\mathbb{R}^n$, and one should note that $\|v_i - v_j\|^2 = \langle v_i, v_i \rangle + \langle v_j, v_j \rangle - 2\langle v_i, v_j \rangle$.

See Lovasz's survey Semidefinite programs and combinatorial optimization for further background and examples.

2 Lifts of polytopes

Consider a polytope $P = \mathbb{R}^n_+ \cap \mathcal{L}$ for some affine subspace $\mathcal{L} \subseteq \mathbb{R}^n$. It can often be uesful to write P as the linear projection of some simpler convex body. As an example, associate to every undirected n-vertex graph $G = (\{1, \ldots, n\}, E)$ the vector $v_G := \mathbf{1}_E \in \{0,1\}^{\binom{n}{2}}$, where $\mathbf{1}_E$ is the indicator vector of its edge set $E \subseteq \binom{[n]}{2}$.

Define ST_n to be the convex hull of vectors v_T as T ranges over all connected trees on $\{1, \ldots, n\}$. This is the spanning tree polytope of the complete graph K_n. One could solve the general minimum spanning tree problem by minimizing $\langle c, x\rangle$ over all $x \in \mathrm{ST}_n$, where $c \in \mathbb{R}^{\binom{n}{2}}$ is the vector denoting the edge weights of the input graph. Unfortunately, it is known that the number of inequalities needed to characterize ST_n is exponential in n, making this optimization problem infeasible for generic methods.

But it turns out there is a polytope $Q \subseteq \mathbb{R}^{\binom{n}{3}}$ that can be specified using only $O(n^3)$ inequalities, and such that Q linearly projects to P. It's not too hard to work out such a polytope; here's a hint: In addition to the variables x_{ij} indicating whether $\{i, j\}$ is an edge in the tree, introduce new variables z_{ikj} that are meant to indicate whether k occurs in the unique i-j path in the tree. Thus one *can,* in polynomial time, solve the minimum spanning tree problem by optimizing instead over Q.

If Q is a polytope and $P = \pi(Q)$ for some linear projection π, one says that Q *is a (polyhedral) lift of* P. The *LP extension complexity* of P is the minimum number d such that P has a polyhedral lift Q that can be defined using only d inequalities. In order to study LP extension complexity, Yannakakis made a fundamental connection with nonnegative factorizations of P's slack matrix.

2.1 Slack matrices

Every polytope P in $\mathbb{R}^n$ can be written in two different ways, as the convex hull of its vertices, or as the intersection of finitely-many halfspaces:

$$P = \mathrm{conv}(x_1, x_2, \ldots, x_k), \tag{2.1}$$

$$P = \{x \in \mathbb{R}^n : \langle a_1, x\rangle \leqslant b_1, \ldots, \langle a_m, x\rangle \leqslant b_m\}. \tag{2.2}$$

To any such pair of representations, we can associate the *slack matrix* $S \in \mathbb{R}^{m\times k}$ given by

$$S_{ij} := b_i - \langle a_i, x_j\rangle.$$

Note that all the entries of S are nonnegative since every vertex $x_i \in P$ is feasible.

Ranks: Normal, nonnegative, and PSD. The *rank* of the matrix S is the smallest dimension d such that one can write

$$S_{ij} = \langle u_i, v_j\rangle, \quad u_1, \ldots, u_m, v_1, \ldots, v_k \in \mathbb{R}^d.$$

The *nonnegative rank* of S is the smallest d such that

$$S_{ij} = \langle u_i, v_j\rangle, \quad u_1, \ldots, u_m, v_1, \ldots, v_k \in \mathbb{R}^d_+.$$

Note the only difference: For nonnegative rank, we have restricted ourself to *nonnegative* vectors. The following theorem gives the connection to LP extension complexity.

Theorem 2.1 (Yannakakis Factorization Theorem)**.** *For any polytope P and any slack matrix S of P, the LP extension complexity of P is equal to the nonnegative rank of P.*

One could rewrite the nonnegative rank as the smallest d such that

$$S_{ij} = \mathrm{Tr}(U_i V_j), \quad U_1, \ldots, U_m, V_1, \ldots, V_k \text{ diagonal matrices in } \mathbb{S}^+_d.$$

By removing the diagonal restriction, we are led to the *positive semidefinite rank of S,* which is the minimum d such that

$$S_{ij} = \mathrm{Tr}(U_i, V_j), \quad U_1, \ldots, U_m, V_1, \ldots, V_k \in \mathbb{S}^+_d. \tag{2.3}$$

2.2 Semidefinite characterizations of polytopes

If one can write $P = \pi(\mathcal{U})$, where π is a linear projection and $\mathcal{U} \subseteq \mathbb{S}_N^+$ is a spectrahedron, one says that *$\mathcal{U}$ is a spectrahedral lift of P*. The *SDP extension complexity of a polytope P* is the minimum value N such that P admits a spectrahedral lift $\mathcal{U} \subseteq \mathbb{S}_N^+$. There is a PSD analog of Yannakakis' Factorzation Theorem.

Theorem 2.2. *For any polytope P, the SDP extension complexity of P is equal to the PSD rank of any slack matrix of P.*

Let's prove this. Consider a polytope P and two representations (2.1) and (2.2). Let A denote the matrix with rows $a_1, \ldots, a_m$, so that $P = \{x \in \mathbb{R}^n : Ax \leqslant b\}$. First, suppose we have a factorization of the form (2.3), and define the spectrahedron

$$Q = \left\{(x, Y) \in \mathbb{R}^n \times \mathbb{S}_d^+ : \langle a_i, x\rangle + \langle U_i, Y\rangle = b_i, i = 1, \ldots, m\right\}.$$

We claim that $P = \pi_n(Q)$, where π_n is the projection of Q to the first n coordinates. Note that $P \subseteq \pi_n(Q)$ because each inner product $\langle U_i, Y\rangle$ is nonnegative, thanks to $U_i, Y \succeq 0$.

Since Q is convex, P is convex, so to establish that $P \supseteq \pi_n(Q)$, it suffices to show that $x_j \in \pi_n(P)$ for every vertex $x_j \in P$. But this follows because $(x_j, V_j) \in Q$, by definition of the factorization (2.3).

The dimension of Q as a spectrahedron is $n + d$, while our goal was to bound the SDP extension complexity by the PSD rank d. Let $\pi_d(Q)$ denote the projection of Q onto the $\mathbb{S}_d$ coordinates. It is an exercise to show that there is a linear map $\pi : \mathbb{S}_d \to \mathbb{R}^n$ that takes $\pi_d(Q)$ to P. (Roughly, for $Y \in \pi_d(Q)$, one defines $\pi(Y) = x$ for some $(x, Y) \in Q$. Since there might be many choices for x, one needs to do this in a consistent way so that the map is linear.) Thus the SDP extension complexity of P is at most the PSD rank of any slack matrix of P.

Nonnegativity proofs. The other, more difficult direction of Theorem 2.2, is instructive especially if you haven't seen this sort of argument before. Suppose $f_1, f_2, \ldots, f_k : \mathbb{R}^d \to \mathbb{R}$ are affine functions, and define

$$\operatorname{cone}(f_1, \ldots, f_k) := \left\{\sum_{i=1}^{k} \lambda_i f_i : \lambda_i \geqslant 0\right\}.$$

Lemma 2.3 (Farkas' Lemma)**.** *Consider a bounded polytope $P = \{x \in \mathbb{R}^n : Ax \leqslant b\}$, where A has rows $a_1, a_2, \ldots, a_d$, and define $f_i(x) := b_i - \langle a_i, x\rangle$. If f is any affine function that is nonnegative on P, then*

$$f \in \operatorname{cone}(f_1, f_2, \ldots, f_d).$$

In other words, the d slack functions $f_1, \ldots, f_d$ generate all the linear inequalities that are true on P. (For a proof of Farkas' Lemma—in the form of an exercise—see Section 2.1 here: Lifts of polytopes and nonnegative rank.)

Consider now a bounded spectrahedron written in the form (1.1):

$$\mathcal{V} = \left\{y \in \mathbb{R}^d : y_1 A_1 + \cdots + y_d A_d \preceq B\right\},$$

for some $A_1, \ldots, A_d, B \in \mathbb{S}_n$.

Lemma 2.4 (PSD Farkas)**.** *Every affine function $f : \mathbb{R}^d \to \mathbb{R}$ that is nonnegative on $\mathcal{V}$ can be written as*

$$f(y) = \operatorname{Tr}\left(V(B - (y_1 A_1 + \cdots + y_d A_d)\right)$$

for some $V \in \mathbb{S}_n^+$.

Proof sketch. By translation, we may assume that $0 \in \mathcal{V}$.

Note that the space $\mathcal{A}_d$ of affine functions $f : \mathbb{R}^d \to \mathbb{R}$ is a $(d+1)$-dimensional vector space. For instance, one can encode f as the vector $(f(e_1), \ldots, f(e_d), f(0))$, where $\{e_1, \ldots, e_d\}$ is the standard basis. We use this to identify $\mathcal{A}_d = \mathbb{R}^{d+1}$.

Define the affine functions

$$f_U(y) := \left\langle U, B - (y_1 A_1 + \cdots + y_d A_d) \right\rangle, \quad U \in \mathbb{S}_n^+$$

and denote

$$\mathcal{V}^* := \left\{ f_U : U \in \mathbb{S}_n^+ \right\} \subseteq \mathcal{A}_d = \mathbb{R}^{d+1}.$$

We will need the following.

Lemma 2.5. *If there is a direction $v \in \mathbb{R}^n$ such that*

$$\sup_{y \in \mathcal{V}} \langle v, (B - (y_1 A_1 + \cdots + y_d A_d)) v \rangle < \infty,$$

then there is a matrix $U \in \mathbb{S}_n^+$ such that

$$\langle U, B - (y_1 A_1 + \cdots + y_d A_d) \rangle = 1, \quad \forall y \in \mathcal{V}.$$

In other words, the constant function $\mathbb{1} \in \mathcal{V}^*$.

Observe that $\mathcal{V}^*$ is a closed[1] convex cone, and

$$y \in \mathcal{V} \iff \left(f(y) \geqslant 0, \ \forall f \in \mathcal{V}^* \right). \tag{2.4}$$

Thus by the hyperplane separation theorem, if $g \in \mathcal{A}_d \setminus \mathcal{V}^*$, it must be that there is a vector $v \in \mathbb{R}^{d+1}$ such that $\langle v, g \rangle < 0$, but $\langle v, f \rangle \geqslant 0$ for all $f \in \mathcal{V}^*$. (Since $\mathcal{V}^*$ is a cone, one can take the separating hyperplane passing through the origin.)

Define $y_v := (v_1, v_2, \ldots, v_d)$ and note that for any $f \in \mathcal{A}_d$,

$$\langle v, f \rangle = v_{d+1} f(0) + v_1 f(e_1) + \cdots + v_d f(e_d) = v_{d+1} f(0) + f(y_v).$$

Since $\mathbb{1} \in \mathcal{V}^*$ and $0 \in \mathcal{V}$, it holds that $v_{d+1} = v_{d+1}\mathbb{1}(0) \geqslant 0$. Hence for every $f \in \mathcal{V}^*$, we have $f(y_v) \geqslant 0$. By (2.4), this implies that $y_v \in \mathcal{V}$.

On the other hand, $\langle v, g \rangle < 0$ implies that either $g(0) < 0$, or $g(y_v) < 0$. In both cases, it must be that g is negative somewhere on $\mathcal{V}$. We conclude that $\mathcal{V}^*$ contains every affine function that is nonnegative on $\mathcal{V}$, completing the proof. □

Note that if $A_1, \ldots, A_d, B$ are diagonal matrices, then Lemma 2.4 specializes to Lemma 2.3, in which case $\mathrm{diag}(V) = (\lambda_1, \ldots, \lambda_n)$ describes the coefficients in the conic combination.

Now suppose that the PSD extension complexity of $P = \{x \in \mathbb{R}^n : Ax \leqslant b\}$ is at most N. Then there is a spectrahedral lift

$$\mathcal{U} = \left\{ (x, y) \in \mathbb{R}^n \times \mathbb{R}^d : \{\langle r_i, x \rangle + \langle s_i, y \rangle = c_i\}, y_1 A_1 + \cdots + y_d A_d \preceq B \right\},$$

with $A_1, \ldots, A_d, B \in \mathbb{S}_N$, and where $P = \pi_n(\mathcal{U})$, with π_n denoting the projection to the first n coordinates.

[1] The fact that $\mathcal{V}^*$ is closed requires verification.

Observe that the inequalities $Ax \leqslant b$ are satisfied on $\mathcal{U}$ since $(x, y) \in \mathcal{U} \implies x \in P$. Define $f_i(x) = b_i - \langle a_i, x\rangle$ (where $a_1, \ldots, a_m$ are the rows of A) and for each vertex $x_j \in P$, let y_j be such that $(x_j, y^{(j)}) \in \mathcal{U}$. Then by Lemma 2.4, for every i, there is a $V_i \in \mathbb{S}_N^+$ such that

$$f_i(x_j) = \mathrm{Tr}\left(V_i \left(B - (y_1^{(j)} A_1 + \cdots + y_d^{(j)} A_d)\right)\right).$$

or, equivalently,

$$S_{ij} = \mathrm{Tr}(V_i Y_j),$$

where $Y_j := B - (y_1^{(j)} A_1 + \cdots + y_d^{(j)} A_d) \succeq 0$. This gives a rank-$N$ PSD factorization of the slack matrix S, completing the proof of Theorem 2.2.

2.3 PSD rank and quantum communication

Consider now any nonnegative matrix $M \in \mathbb{R}_+^{\mathcal{X}\times\mathcal{Y}}$, where $\mathcal{X}, \mathcal{Y}$ are finite index sets. Define the *PSD rank of M* by

$$\mathrm{rk}_{\mathsf{psd}}(M) := \min\left\{d \geqslant 0 : M_{xy} = \langle P_x, Q_y\rangle \ \forall (x, y) \in \mathcal{X}\times\mathcal{Y}, \text{ for some } \{P_x, Q_y\} \subseteq \mathbb{S}_d^+\right\}.$$

Let us now take a slightly different perspective. Consider two parties Alice and Bob. Alice receives an input $x \in \mathcal{X}$ and Bob receivs $y \in \mathcal{Y}$. Their goal is to communicate (random) messages which we denote by the transcript π_{xy}, and then each output a (random) nonnegative number $\boldsymbol{A}(x, \pi)$ and $\boldsymbol{B}(y, \pi)$, with the constraint that

$$\mathbb{E}\left[\boldsymbol{A}(x, \pi_{xy})\boldsymbol{B}(y, \pi_{xy})\right] = M(x, y), \quad \forall (x, y) \in \mathcal{X}\times\mathcal{Y}.$$

This is called *communication complexity in expectation* and, generally speaking, the cost of the protocol is given by $\max_{x\in\mathcal{X}, y\in\mathcal{Y}} |\pi_{xy}|$, where $|\pi_{xy}|$ is the length of the communication transcript in bits.

It is interesting to note that we can actually assume the protocol is one-way. To achieve this, fix some randomized protocol for Alice and Bob, and let's convert it to a one-way protocol. Alice simulates the protocol, but simply *guesses Bob's responses uniformly at random.* (For simplicity, let's assume a standardized protocol where each player sends one bit in every round.) She then sends the entire faked transcript $\tilde{\pi}$ to Bob, and she outputs $\boldsymbol{A}(x, \tilde{\pi})$. Bob examines the transcript and either outputs 0 if he would have deviated from $\tilde{\pi}$, or outputs $\boldsymbol{B}(y, \tilde{\pi})/p$, where p is the probability of Alice correctly guessing his side of the transcript.

Quantum communication (in expectation). Suppose now that we allow Alice to send Bob a *quantum message.* As we know, this message can be described by a density matrix $\rho_x \in \mathcal{D}(\mathbb{C}^d)$. Bob will then make a measurement of ρ_x and, based on the outcome of this measurement, will output a random nonnegative number, with the goal still being to have the expected output equal $M(x, y)$.

Suppose that $\mathrm{rk}_{\mathsf{psd}}(M) \leqslant d$ so that $M(x, y) = \langle P_x, Q_y\rangle$. By scaling, we may assume that $\mathrm{Tr}(P_x) = 1$ for every $x \in \mathcal{X}$. (Note that scaling $M(x, y)$ is fine, since Bob can always scale his output by a fixed positive number.)

Consider then the following protocol: Alice comunicates according to the density matrix P_x. In other words, write $P_x = \sum_{i=1}^d \lambda_i v_i v_i^*$. Then with probability λ_i, Alice sends the (pure) state v_i to Bob. Bob then outputs $\mathrm{Tr}(Q_y v_i v_i^*)$. He can do this as follows: Let $\alpha = \|Q_y\|$ be the maximum eigenvalue of Q_y so that $\alpha^{-1}Q_y + (I - \alpha^{-1}Q_y) = I$ gives Bob a two-outcome measurement. On obtaining the second outcome, he outputs 0. Upon obtaining the first outcome, he outputs α. His expected output is then $\mathrm{Tr}(Q_y v_i v_i^*)$, and the expected output of the entire protocol is $\mathrm{Tr}(P_x Q_y)$.

We have just shown that if $\mathrm{rk}_{\mathsf{psd}}(M) \leqslant d$, then there is a quantum communication protocol to compute $M(x, y)$ in expectation using at most $\lceil \log_2 d \rceil$ qubits. We leave the converse as an exercise: Given such a communication protocol with q qubits, it holds that $\mathrm{rk}_{\mathsf{psd}}(M) \leqslant 2^q$.