16 Introduction to Semidefinite Programming (SDP)

16.1 Introduction

Semidefinite programming (SDP) is probably the most exciting development in mathematical programming in the last ten years. SDP has applications in such diverse fields as traditional convex constrained optimization, control theory, and combinatorial optimization. Because SDP is solvable via interior-point methods (and usually requires about the same amount of computational resources as linear optimization), most of these applications can usually be solved fairly efficiently in practice as well as in theory.

16.2 A Slightly Different View of Linear Programming

Consider the linear programming problem in standard form:

$$\begin{array}{lll} LP: & \text{minimize} & c \cdot x \\ & \text{s.t.} & a_i \cdot x = b_i, \quad i = 1, \ldots, m \\ & & x \in \mathbb{R}^n_+. \end{array}$$

Here x is a vector of n variables, and we write "$c \cdot x$" for the inner-product "$\sum_{j=1}^n c_j x_j$", etc.

Also, $\mathbb{R}^n_+ := \{x \in \mathbb{R}^n \mid x \geq 0\}$, and we call $\mathbb{R}^n_+$ the nonnegative orthant. In fact, $\mathbb{R}^n_+$ is a *closed convex cone*, where K is called a closed a convex cone if K satisfies the following two conditions:

- If $x, w \in K$, then $\alpha x + \beta w \in K$ for all nonnegative scalars α and β.
- K is a closed set.

In words, LP is the following problem:

"Minimize the linear function $c \cdot x$, subject to the condition that x must solve m given equations $a_i \cdot x = b_i$, $i = 1, \ldots, m$, and that x must lie in the closed convex cone $K = \mathbb{R}^n_+$."

We will write the standard linear programming dual problem as:

$$\begin{array}{lll} LD: & \text{maximize} & \sum_{i=1}^m y_i b_i \\ & \text{s.t.} & \sum_{i=1}^m y_i a_i + s = c \\ & & s \in \mathbb{R}^n_+. \end{array}$$

Given a feasible solution x of LP and a feasible solution (y, s) of LD, the duality gap is simply $c \cdot x - \sum_{i=1}^m y_i b_i = (c - \sum_{i=1}^m y_i a_i) \cdot x = s \cdot x \geq 0$, because $x \geq 0$ and $s \geq 0$. We know from LP duality theory that so long as the primal problem LP is feasible and has bounded optimal objective value, then the primal and the dual both attain their optima with no duality gap. That is, there exists x^* and (y^*, s^*) feasible for the primal and dual, respectively, for which $c \cdot x^* - \sum_{i=1}^m y_i^* b_i = s^* \cdot x^* = 0$.

16.3 Facts about Matrices and the Semidefinite Cone

16.3.1 Facts about the Semidefinite Cone

If X is an $n \times n$ matrix, then X is a symmetric positive semidefinite (SPSD) matrix if $X = X^T$ and

$$v^T X v \geq 0 \text{ for any } v \in \mathbb{R}^n.$$

If X is an $n \times n$ matrix, then X is a symmetric positive definite (SPD) matrix if $X = X^T$ and

$$v^T X v > 0 \text{ for any } v \in \mathbb{R}^n, v \neq 0.$$

Let S^n denote the set of symmetric $n \times n$ matrices, and let S^n_+ denote the set of symmetric positive semidefinite (SPSD) $n \times n$ matrices. Similarly let S^n_{++} denote the set of symmetric positive definite (SPD) $n \times n$ matrices.

Let X and Y be any symmetric matrices. We write "$X \succeq 0$" to denote that X is SPSD, and we write "$X \succeq Y$" to denote that $X - Y \succeq 0$. We write "$X \succ 0$" to denote that X is SPD, etc.

$S^n_+ = \{X \in S^n \mid X \succeq 0\}$ is a closed convex cone in $\mathbb{R}^{n^2}$ of dimension $n \times (n+1)/2$.

To see why this remark is true, suppose that $X, W \in S^n_+$. Pick any scalars $\alpha, \beta \geq 0$. For any $v \in \mathbb{R}^n$, we have:

$$v^T(\alpha X + \beta W)v = \alpha v^T X v + \beta v^T W v \geq 0,$$

whereby $\alpha X + \beta W \in S^n_+$. This shows that S^n_+ is a cone. It is also straightforward to show that S^n_+ is a closed set.

16.3.2 Facts about Eigenvalues and Eigenvectors

If M is a square $n \times n$ matrix, then λ is an eigenvalue of M with corresponding eigenvector x if

$$Mx = \lambda x \text{ and } x \neq 0.$$

Note that λ is an eigenvalue of M if and only if λ is a root of the polynomial:

$$p(\lambda) := \det(M - \lambda I),$$

that is

$$p(\lambda) = \det(M - \lambda I) = 0.$$

This polynomial will have n roots counting multiplicities, that is, there exist $\lambda_1, \lambda_2, \ldots, \lambda_n$ for which:

$$p(\lambda) := \det(M - \lambda I) = \Pi_{i=1}^n (\lambda_i - \lambda).$$

If M is symmetric, then all eigenvalues λ of M must be real numbers, and these eigenvalues can be ordered so that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ if we so choose.

The corresponding eigenvectores $q^1, \ldots, q^n$ of M can be chosen so that they are orthogonal, namely $\left(q^i\right)^T \left(q^j\right) = 0$ for $i \neq j$, and can be scaled so that $\left(q^i\right)^T \left(q^i\right) = 1$. This means that the matrix:

$$Q := \left[q^1 \ q^2 \ \cdots \ q^n\right]$$

satisfies:

$$Q^T Q = I,$$

or put another way:

$$Q^T = Q^{-1}.$$

We call such a matrix *orthonormal*.

Let us assemble the ordered eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ into a diagonal matrix D:

$$D := \begin{pmatrix} \lambda_1 & 0 & & 0 \\ 0 & \lambda_2 & & 0 \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix}.$$

Then we have:

Property: $M = QDQ^T$. To prove this, notice that $MQ = QD$, and so post-multiplying by Q^T yields: $M = MQQ^T = QDQ^T$.

The decomposition of M into $M = QDQ^T$ is called its *eigendecomposition*.

16.3.3 Facts about symmetric matrices

- If $X \in S^n$, then $X = QDQ^T$ for some orthonormal matrix Q and some diagonal matrix D. (Recall that Q is orthonormal means that $Q^{-1} = Q^T$, and that D is diagonal means that the off-diagonal entries of D are all zeros.)
- If $X = QDQ^T$ as above, then the columns of Q form a set of n orthogonal eigenvectors of X, whose eigenvalues are the corresponding entries of the diagonal matrix D.
- $X \succeq 0$ if and only if $X = QDQ^T$ where the eigenvalues (i.e., the diagonal entries of D) are all nonnegative.
- $X \succ 0$ if and only if $X = QDQ^T$ where the eigenvalues (i.e., the diagonal entries of D) are all positive.
- If M is symmetric, then $\det(M) = \Pi_{j=1}^n \lambda_j$.
- If $X \succeq 0$ then $X_{ii} \geq 0, i = 1, \ldots, n$.
- If $X \succeq 0$ and if $X_{ii} = 0$, then $X_{ij} = X_{ji} = 0$ for all $j = 1, \ldots, n$.
- Consider the matrix M defined as follows:

  $$M = \begin{pmatrix} P & v \\ v^T & d \end{pmatrix},$$

  where $P \succ 0$, v is a vector, and d is a scalar. Then $M \succeq 0$ if and only if $d - v^T P^{-1} v \geq 0$.
- For a given column vector a, the matrix $X := aa^T$ is SPSD, i.e., $X = aa^T \succeq 0$.

Also note the following:

- If $M \succeq 0$, then there is a matrix N for which $M = N^T N$. To see this, simply take $N = D^{\frac{1}{2}} Q^T$.
- If M is symmetric, then $\sum_{j=1}^n M_{jj} = \sum_{j=1}^n \lambda_j$

16.4 Semidefinite Programming

Let $X \in S^n$. We can think of X as a matrix, or equivalently, as an array of n^2 components of the form $(x_{11}, \ldots, x_{nn})$. We can also just think of X as an object (a vector) in the space S^n. All three different equivalent ways of looking at X will be useful.

What will a linear function of X look like? If $C(X)$ is a linear function of X, then $C(X)$ can be written as $C \bullet X$, where

$$C \bullet X := \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} X_{ij}.$$

If X is a symmetric matrix, there is no loss of generality in assuming that the matrix C is also symmetric. With this notation, we are now ready to define a semidefinite program. A semidefinite program (SDP) is an optimization problem of the form:

$$\begin{array}{lll} SDP: & \text{minimize} & C \bullet X \\ & \text{s.t.} & A_i \bullet X = b_i, \ i = 1, \ldots, m, \\ & & X \succeq 0. \end{array}$$

Notice that in an SDP that the variable is the matrix X, but it might be helpful to think of X as an array of n^2 numbers or simply as a vector in S^n. The objective function is the linear function $C \bullet X$ and there are m linear equations that X must satisfy, namely $A_i \bullet X = b_i, \ i = 1, \ldots, m$. The variable X also must lie in the (closed convex) cone of positive semidefinite symmetric matrices S^n_+. Note that the data for SDP consists of the symmetric matrix C (which is the data for the objective function) and the m symmetric matrices $A_1, \ldots, A_m$, and the $m-$vector b, which form the m linear equations.

Let us see an example of an SDP for $n = 3$ and $m = 2$. Define the following matrices:

$$A_1 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 3 & 7 \\ 1 & 7 & 5 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0 & 2 & 8 \\ 2 & 6 & 0 \\ 8 & 0 & 4 \end{pmatrix}, \ b = \begin{pmatrix} 11 \\ 19 \end{pmatrix}, \ \text{and } C = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 9 & 0 \\ 3 & 0 & 7 \end{pmatrix}.$$

Then the variable X will be the 3×3 symmetric matrix:

$$X = \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix},$$

and so, for example,

$$\begin{aligned} C \bullet X &= x_{11} + 2x_{12} + 3x_{13} + 2x_{21} + 9x_{22} + 0x_{23} + 3x_{31} + 0x_{32} + 7x_{33} \\ &= x_{11} + 4x_{12} + 6x_{13} + 9x_{22} + 0x_{23} + 7x_{33}. \end{aligned}$$

since, in particular, X is symmetric. Therefore the SDP can be written as:

$$\begin{array}{llrcl} SDP: & \text{minimize} & x_{11} + 4x_{12} + 6x_{13} + 9x_{22} + 0x_{23} + 7x_{33} & & \\ & \text{s.t.} & x_{11} + 0x_{12} + 2x_{13} + 3x_{22} + 14x_{23} + 5x_{33} & = & 11 \\ & & 0x_{11} + 4x_{12} + 16x_{13} + 6x_{22} + 0x_{23} + 4x_{33} & = & 19 \\ & & X = \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix} \succeq 0. & & \end{array}$$

Notice that SDP looks remarkably similar to a linear program. However, the standard LP constraint that x must lie in the nonnegative orthant is replaced by the constraint that the variable X must lie in the cone of positive semidefinite matrices. Just as "$x \geq 0$" states that each of the n components of x must be nonnegative, it may be helpful to think of "$X \succeq 0$" as stating that each of the n *eigenvalues* of X must be nonnegative. It is easy to see that a linear program LP is a special instance of an SDP. To see one way of doing this, suppose that $(c, a_1, \ldots, a_m, b_1, \ldots, b_m)$ comprise the data for LP. Then define:

$$A_i = \begin{pmatrix} a_{i1} & 0 & \ldots & 0 \\ 0 & a_{i2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & a_{in} \end{pmatrix}, \quad i = 1, \ldots, m, \text{ and } C = \begin{pmatrix} c_1 & 0 & \ldots & 0 \\ 0 & c_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & c_n \end{pmatrix}.$$

Then LP can be written as:

$$\begin{array}{lll} SDP: & \text{minimize} & C \bullet X \\ & \text{s.t.} & A_i \bullet X = b_i, \ i = 1, \ldots, m, \\ & & X_{ij} = 0, \quad i = 1, \ldots, n, \quad j = i+1, \ldots, n, \\ & & X \succeq 0, \end{array}$$

with the association that

$$X = \begin{pmatrix} x_1 & 0 & \ldots & 0 \\ 0 & x_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & x_n \end{pmatrix}.$$

Of course, in practice one would never *want* to convert an instance of LP into an instance of SDP. The above construction merely shows that SDP includes linear programming as a special case.

16.5 Semidefinite Programming Duality

The dual problem of SDP is defined (or derived from first principles) to be:

$$\begin{array}{lll} SDD: & \text{maximize} & \sum_{i=1}^{m} y_i b_i \\ & \text{s.t.} & \sum_{i=1}^{m} y_i A_i + S = C \\ & & S \succeq 0. \end{array}$$

One convenient way of thinking about this problem is as follows. Given multipliers $y_1, \ldots, y_m$, the objective is to maximize the linear function $\sum_{i=1}^{m} y_i b_i$. The constraints of SDD state that the matrix S defined as

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

must be positive semidefinite. That is,

$$C - \sum_{i=1}^{m} y_i A_i \succeq 0.$$

We illustrate this construction with the example presented earlier. The dual problem is:

$$\begin{array}{rll} SDD: & \text{maximize} & 11y_1 + 19y_2 \\ & \text{s.t.} & y_1 \begin{pmatrix} 1 & 0 & 1 \\ 0 & 3 & 7 \\ 1 & 7 & 5 \end{pmatrix} + y_2 \begin{pmatrix} 0 & 2 & 8 \\ 2 & 6 & 0 \\ 8 & 0 & 4 \end{pmatrix} + S = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 9 & 0 \\ 3 & 0 & 7 \end{pmatrix} \\ & & S \succeq 0, \end{array}$$

which we can rewrite in the following form:

$$\begin{array}{rll} SDD: & \text{maximize} & 11y_1 + 19y_2 \\ & \text{s.t.} & \\ & & \begin{pmatrix} 1 - 1y_1 - 0y_2 & 2 - 0y_1 - 2y_2 & 3 - 1y_1 - 8y_2 \\ 2 - 0y_1 - 2y_2 & 9 - 3y_1 - 6y_2 & 0 - 7y_1 - 0y_2 \\ 3 - 1y_1 - 8y_2 & 0 - 7y_1 - 0y_2 & 7 - 5y_1 - 4y_2 \end{pmatrix} \succeq 0. \end{array}$$

It is often easier to "see" and to work with a semidefinite program when it is presented in the format of the dual SDD, since the variables are the m multipliers $y_1, \ldots, y_m$.

As in linear programming, we can switch from one format of SDP (primal or dual) to any other format with great ease, and there is no loss of generality in assuming a particular specific format for the primal or the dual.

The following proposition states that weak duality must hold for the primal and dual of SDP:

Proposition 91 *Given a feasible solution X of SDP and a feasible solution (y, S) of SDD, the duality gap is $C \bullet X - \sum_{i=1}^m y_i b_i = S \bullet X \geq 0$. If $C \bullet X - \sum_{i=1}^m y_i b_i = 0$, then X and (y, S) are each optimal solutions to SDP and SDD, respectively, and furthermore, $SX = 0$.*

In order to prove Proposition 91, it will be convenient to work with the *trace* of a matrix, defined below:

$$\operatorname{trace}(M) = \sum_{j=1}^n M_{jj}.$$

Simple arithmetic can be used to establish the following two elementary identifies:

Property: $A \bullet B = \operatorname{trace}(A^T B)$. To prove this, notice that $\operatorname{trace}(A^T B) = \sum_{j=1}^n \left(A^T B\right)_{jj} = \sum_{j=1}^n \left(\sum_{i=1}^n A_{ij} B_{ij}\right) = A \bullet B$.

Property: $\operatorname{trace}(MN) = \operatorname{trace}(NM)$. To prove this, simply notice that $\operatorname{trace}(MN) = M^T \bullet N = \sum_{i=1}^n \sum_{j=1}^n M_{ji} N_{ij} = \sum_{i=1}^n \sum_{j=1}^n N_{ij} M_{ji} = \sum_{i=1}^n \sum_{j=1}^n N_{ji} M_{ij} = N^T \bullet M = \operatorname{trace}(NM)$.

Proof of Proposition 91. For the first part of the proposition, we must show that if $S \succeq 0$ and $X \succeq 0$, then $S \bullet X \geq 0$. Let $S = PDP^T$ and $X = QEQ^T$ where P, Q are orthonormal matrices and D, E are nonnegative diagonal matrices. We have:

$$S \bullet X = \operatorname{trace}(S^T X) = \operatorname{trace}(SX) = \operatorname{trace}(PDP^T QEQ^T)$$

$$= \operatorname{trace}(DP^T QEQ^T P) = \sum_{j=1}^n D_{jj} (P^T QEQ^T P)_{jj} \geq 0,$$

where the last inequality follows from the fact that all $D_{jj} \geq 0$ and the fact that the diagonal of the symmetric positive semidefinite matrix $P^T QEQ^T P$ must be nonnegative.

To prove the second part of the proposition, suppose that trace$(SX) = 0$. Then from the above equalities, we have

$$\sum_{j=1}^{n} D_{jj}(P^T QEQ^T P)_{jj} = 0.$$

However, this implies that for each $j = 1, \ldots, n$, either $D_{jj} = 0$ or the $(P^T QEQ^T P)_{jj} = 0$. Furthermore, the latter case implies that the j^{th} row of $P^T QEQ^T P$ is all zeros. Therefore $DP^T QEQ^T P = 0$, and so $SX = PDP^T QEQ^T = 0$. ∎

Unlike the case of linear programming, we cannot assert that either SDP or SDD will attain their respective optima, and/or that there will be no duality gap, unless certain regularity conditions hold. One such regularity condition which ensures that strong duality will prevail is a version of the "Slater condition," summarized in the following theorem which we will not prove:

Theorem 92 *Let z_P^* and z_D^* denote the optimal objective function values of SDP and SDD, respectively. Suppose that there exists a feasible solution $\hat{X}$ of SDP such that $\hat{X} \succ 0$, and that there exists a feasible solution $(\hat{y}, \hat{S})$ of SDD such that $\hat{S} \succ 0$. Then both SDP and SDD attain their optimal values, and $z_P^* = z_D^*$.*

16.6 Key Properties of Linear Programming that do not extend to SDP

The following summarizes some of the more important properties of linear programming that do not extend to SDP:

- There may be a finite or infinite duality gap. The primal and/or dual may or may not attain their optima. However, as noted above in Theorem 92, both programs will attain their common optimal value if both programs have feasible solutions that are SPD.
- There is no finite algorithm for solving SDP. There is a simplex algorithm, but it is not a finite algorithm. There is no direct analog of a "basic feasible solution" for SDP.

16.7 SDP in Combinatorial Optimization

SDP has wide applicability in combinatorial optimization. A number of $NP-$hard combinatorial optimization problems have convex relaxations that are semidefinite programs. In many instances, the SDP relaxation is very tight in practice, and in certain instances in particular, the optimal solution to the SDP relaxation can be converted to a feasible solution for the original problem with provably good objective value. An example of the use of SDP in combinatorial optimization is given below.

16.7.1 An SDP Relaxation of the MAX CUT Problem

Let G be an undirected graph with nodes $N = \{1, \ldots, n\}$, and edge set E. Let $w_{ij} = w_{ji}$ be the weight on edge (i, j), for $(i, j) \in E$. We assume that $w_{ij} \geq 0$ for all $(i, j) \in E$. The MAX CUT problem is to determine a subset S of the nodes N for which the sum of the weights of the edges that cross from S to its complement $\bar{S}$ is maximized (where $\bar{S} := N \setminus S$).

We can formulate MAX CUT as an integer program as follows. Let $x_j = 1$ for $j \in S$ and $x_j = -1$ for $j \in \bar{S}$. Then our formulation is:

$$\begin{array}{rll} MAXCUT: & \text{maximize}_x & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1 - x_i x_j) \\ & \text{s.t.} & x_j \in \{-1, 1\}, \quad j = 1, \ldots, n. \end{array}$$

Now let

$$Y = xx^T,$$

whereby

$$Y_{ij} = x_i x_j, \ i = 1, \ldots, n, \quad j = 1, \ldots, n.$$

Also let W be the matrix whose $(i,j)^{\text{th}}$ element is w_{ij} for $i = 1, \ldots, n$ and $j = 1, \ldots, n$. Then MAX CUT can be equivalently formulated as:

$$\begin{array}{rll} MAXCUT: & \text{maximize}_{Y,x} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} - \frac{1}{4} W \bullet Y \\ & \text{s.t.} & x_j \in \{-1, 1\}, \quad j = 1, \ldots, n \\ & & Y = xx^T. \end{array}$$

Notice in this problem that the first set of constraints are equivalent to $Y_{jj} = 1, \ j = 1, \ldots, n$. We therefore obtain:

$$\begin{array}{rll} MAXCUT: & \text{maximize}_{Y,x} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} - \frac{1}{4} W \bullet Y \\ & \text{s.t.} & Y_{jj} = 1, \quad j = 1, \ldots, n \\ & & Y = xx^T. \end{array}$$

Last of all, notice that the matrix $Y = xx^T$ is a symmetric rank-1 positive semidefinite matrix. If we *relax* this condition by removing the rank-1 restriction, we obtain the following relaxtion of MAX CUT, which is a semidefinite program:

$$\begin{array}{rll} RELAX: & \text{maximize}_{Y} & \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} - \frac{1}{4} W \bullet Y \\ & \text{s.t.} & Y_{jj} = 1, \quad j = 1, \ldots, n \\ & & Y \succeq 0. \end{array}$$

It is therefore easy to see that RELAX provides an upper bound on MAXCUT, i.e.,

$$MAXCUT \leq RELAX.$$

As it turns out, one can also prove without too much effort that:

$$0.87856 \ RELAX \leq MAXCUT \leq RELAX.$$

This is an impressive result, in that it states that the value of the semidefinite relaxation is guaranteed to be no more than 12.2% higher than the value of NP-hard problem MAX CUT.

16.8 SDP in Convex Optimization

As stated above, SDP has very wide applications in convex optimization. The types of constraints that can be modelled in the SDP framework include: linear inequalities, convex quadratic inequalities, lower bounds on matrix norms, lower bounds on determinants of SPSD matrices, lower bounds on the geometric mean of a nonnegative vector, plus many others. Using these and other constructions, the following problems (among many others) can be cast in the form of a semidefinite program: linear programming, optimizing a convex quadratic form subject to convex quadratic inequality constraints, minimizing the volume of an ellipsoid that covers a given set of points and ellipsoids, maximizing the volume of an ellipsoid that is contained in a given polytope, plus a variety of maximum eigenvalue and minimum eigenvalue problems. In the subsections below we demonstrate how some important problems in convex optimization can be re-formulated as instances of SDP.

16.8.1 SDP for Convex Quadratically Constrained Quadratic Programming

A convex quadratically constrained quadratic program is a problem of the form:

$$\begin{array}{lll} QCQP: & \underset{x}{\text{minimize}} & x^TQ_0x + q_0^Tx + c_0 \\ & \text{s.t.} & x^TQ_ix + q_i^Tx + c_i \le 0, \ i = 1, \ldots, m, \end{array}$$

where the $Q_0 \succeq 0$ and $Q_i \succeq 0, \quad i = 1, \ldots, m$. This problem is the same as:

$$\begin{array}{lll} QCQP: & \underset{x,\theta}{\text{minimize}} & \theta \\ & \text{s.t.} & x^TQ_0x + q_0^Tx + c_0 - \theta \le 0 \\ & & x^TQ_ix + q_i^Tx + c_i \le 0, \ i = 1, \ldots, m. \end{array}$$

We can factor each Q_i into

$$Q_i = M_i^T M_i$$

for some matrix M_i. Then note the equivalence:

$$\begin{pmatrix} I & M_ix \\ x^TM_i^T & -c_i - q_i^Tx \end{pmatrix} \succeq 0 \quad \iff \quad x^TQ_ix + q_i^Tx + c_i \le 0.$$

In this way we can write $QCQP$ as:

$$\begin{array}{lll} QCQP: & \underset{x,\theta}{\text{minimize}} & \theta \\ & \text{s.t.} & \\ & & \begin{pmatrix} I & M_0x \\ x^TM_0^T & -c_0 - q_0^Tx + \theta \end{pmatrix} \succeq 0 \\ & & \begin{pmatrix} I & M_ix \\ x^TM_i^T & -c_i - q_i^Tx \end{pmatrix} \succeq 0, \ i = 1, \ldots, m. \end{array}$$

Notice in the above formulation that the variables are θ and x and that all matrix coefficients are linear functions of θ and x.

16.8.2 SDP for Second-Order Cone Optimization

A second-order cone optimization problem (SOCP) is an optimization problem of the form:

$$\begin{array}{lll} \text{SOCP:} & \min_x & c^T x \\ & \text{s.t.} & Ax = b \\ & & \|Q_i x + d_i\| \le \left(g_i^T x + h_i\right) \ , \quad i = 1, \ldots, k. \end{array}$$

In this problem, the norm $\|v\|$ is the standard Euclidean norm:

$$\|v\| := \sqrt{v^T v}.$$

The norm constraints in SOCP are called "second-order cone" constraints. Note that these are convex constraints.

Here we show that any second-order cone constraint can be written as an SDP constraint. Indeed we have:

Property:

$$\|Qx + d\| \le \left(g^T x + h\right) \iff \begin{pmatrix} (g^T x + h)I & (Qx + d) \\ (Qx + d)^T & g^T x + h \end{pmatrix} \succeq 0.$$

Note in the above that the matrix involved here is a linear function of the variable x, and so is in the general form of an SDP constraint. This property is a direct consequence of the fact (stated earlier) that

$$M = \begin{pmatrix} P & v \\ v^T & d \end{pmatrix} \succeq 0 \iff d - v^T P^{-1} v \ge 0.$$

Therefore we can write the second-order cone optimization problem as:

$$\begin{array}{lll} \text{SDPSOCP:} & \min_x & c^T x \\ & \text{s.t.} & Ax = b \\ & & \begin{pmatrix} (g_i^T x + h_i)I & (Q_i x + d_i) \\ (Q_i x + d_i)^T & g_i^T x + h_i \end{pmatrix} \succeq 0 \ , \quad i = 1, \ldots, k. \end{array}$$

16.8.3 SDP for Eigenvalue Optimization

There are many types of eigenvalue optimization problems that can be formulated as SDPs. In a typical eigenvalue optimization problem, we are given symmetric matrices B and A_i, $i = 1, \ldots, k$, and we choose weights $w_1, \ldots, w_k$ to create a new matrix S:

$$S := B - \sum_{i=1}^{k} w_i A_i.$$

In some applications there might be restrictions on the weights w, such as $w \ge 0$ or more generally linear inequalities of the form $Gw \le d$. The typical goal is then to choose w in such a way that the eigenvalues of S are "well-aligned," for example:

- $\lambda_{\min}(S)$ is maximized
- $\lambda_{\max}(S)$ is minimized

- $\lambda_{\max}(S) - \lambda_{\min}(S)$ is minimized
- $\sum_{j=1}^{n} \lambda_j(S)$ is minimized or maximized

Let us see how to work with these problems using SDP. First, we have:

Property: $M \succeq tI$ if and only if $\lambda_{\min}(M) \geq t$.

To see why this is true, let us consider the eigenvalue decomposition of $M = QDQ^T$, and consider the matrix R defined as:

$$R = M - tI = QDQ^T - tI = Q(D - tI)Q^T.$$

Then

$$M \succeq tI \iff R \succeq 0 \iff D - tI \succeq 0 \iff \lambda_{\min}(M) \geq t.$$

Property: $M \preceq tI$ if and only if $\lambda_{\max}(M) \leq t$.

To see why this is true, let us consider the eigenvalue decomposition of $M = QDQ^T$, and consider the matrix R defined as:

$$R = M - tI = QDQ^T - tI = Q(D - tI)Q^T.$$

Then

$$M \preceq tI \iff R \preceq 0 \iff D - tI \preceq 0 \iff \lambda_{\max}(M) \leq t.$$

Now suppose that we wish to find weights w to minimize the difference between the largest and the smallest eigenvalues of S. This problem can be written down as:

$$\begin{array}{rll} EOP: & \underset{w,S}{\text{minimize}} & \lambda_{\max}(S) - \lambda_{\min}(S) \\ & \text{s.t.} & S = B - \sum_{i=1}^{k} w_i A_i \\ & & Gw \leq d. \end{array}$$

Then EOP can be written as:

$$\begin{array}{rll} EOP: & \underset{w,S,\mu,\lambda}{\text{minimize}} & \mu - \lambda \\ & \text{s.t.} & S = B - \sum_{i=1}^{k} w_i A_i \\ & & Gw \leq d \\ & & \lambda I \preceq S \preceq \mu I. \end{array}$$

This last problem is a semidefinite program.

Using constructs such as those shown above, very many other types of eigenvalue optimization problems can be formulated as SDPs. For example, suppose that we would like to work with $\sum_{j=1}^{n} \lambda_j(S)$. Then one can use elementary properties of the determinant function to prove:

Property: If M is symmetric, then $\sum_{j=1}^{n} \lambda_j(S) = \sum_{j=1}^{n} M_{jj}$.

Then we can work with $\sum_{j=1}^{n} \lambda_j(S)$ by using instead $I \bullet S$. Therefore enforcing a constraint that the sum of the eigenvalues must lie between l and u can be written as:

$$
\begin{array}{lll}
EOP2: & \displaystyle\operatorname*{minimize}_{w,S,\mu,\lambda} & \mu - \lambda \\
& \text{s.t.} & S = B - \sum_{i=1}^{k} w_i A_i \\
& & Gw \leq d \\
& & \lambda I \preceq S \preceq \mu I \\
& & l \leq I \bullet S \leq u.
\end{array}
$$

This last problem is a semidefinite program.

16.8.4 The Logarithmic Barrier Function

At the heart of an interior-point method is a barrier function that exerts a repelling force from the boundary of the feasible region. For SDP, we need a barrier function whose values approach $+\infty$ as points X approach the boundary of the semidefinite cone S_+^n.

Let $X \in S_+^n$. Then X will have n eigenvalues, say $\lambda_1(X), \ldots, \lambda_n(X)$ (possibly counting multiplicities). We can characterize the boundary of the semidefinite cone as follows:

$$\partial S_+^n = \{X \in S^n \mid \lambda_j(X) \geq 0, j = 1, \ldots, n, \text{ and } \lambda_j(X) = 0 \text{ for some } j \in \{1, \ldots, n\}\}.$$

A natural barrier function to use to repel X from the boundary of S_+^n then is

$$B(X) := -\sum_{j=1}^{n} \ln(\lambda_i(X)) = -\ln(\prod_{j=1}^{n} \lambda_i(X)) = -\ln(\det(X)).$$

This function is called the log-determinant function or the logarithmic barrier function for the semidefinite cone. It is not too difficult to derive the gradient and the Hessian of $B(X)$ and to construct the following quadratic Taylor expansion of $B(X)$:

$$B(\bar{X} + \alpha S) \approx B(\bar{X}) + \alpha \bar{X}^{-1} \bullet S + \frac{1}{2}\alpha^2 \left(\bar{X}^{-\frac{1}{2}} S \bar{X}^{-\frac{1}{2}}\right) \bullet \left(\bar{X}^{-\frac{1}{2}} S \bar{X}^{-\frac{1}{2}}\right).$$

The barrier function $B(X)$ has the same remarkable properties in the context of interior-point methods for SDP as the barrier function $-\sum_{j=1}^{n} \ln(x_j)$ does in the context of linear optimization.

16.8.5 The Analytic Center Problem for SDP

Just as in linear optimization, we can consider the analytic center problem for SDP. Given a system of the form:

$$\sum_{i=1}^{m} y_i A_i \preceq C,$$

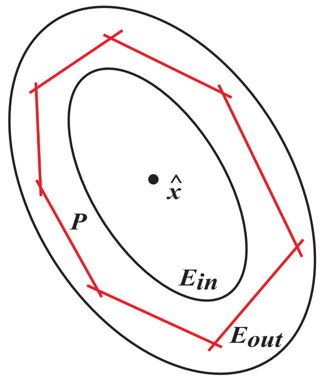

Figure 5: Illustration of the ellipsoid construction at the analytic center.

the *analytic center* is the solution $(\hat{y}, \hat{S})$ of the following optimization problem:

$$\begin{array}{lll} \text{(ACP:)} & \text{maximize}_{y,S} & \prod_{i=1}^{n} \lambda_i(S) \\ & \text{s.t.} & \sum_{i=1}^{m} y_i A_i + S = C \\ & & S \succeq 0. \end{array}$$

This is easily seen to be the same as:

$$\begin{array}{lll} \text{(ACP:)} & \text{minimize}_{y,S} & -\ln\det(S) \\ & \text{s.t.} & \sum_{i=1}^{m} y_i A_i + S = C \\ & & S \succ 0. \end{array}$$

Just as in linear inequality systems, the analytic center possesses a very nice "centrality" property in the feasible region P of the semi-definite inequality system. Suppose that $(\hat{y}, \hat{S})$ is the analytic center. Then there are easy-to-construct ellipsoids E_{IN} and E_{OUT}, both centered at $\hat{y}$ and where E_{OUT} is a scaled version of E_{IN} with scale factor n, with the property that:

$$E_{\text{IN}} \subset P \subset E_{\text{OUT}},$$

as illustrated in Figure 5.

16.8.6 SDP for the Minimum Volume Circumscription Problem

A given matrix $R \succ 0$ and a given point z can be used to define an ellipsoid in $\mathbb{R}^n$:

$$E_{R,z} := \{y \mid (y-z)^T R (y-z) \le 1\}.$$

One can prove that the volume of $E_{R,z}$ is proportional to $\sqrt{\det(R^{-1})}$.

Suppose we are given a convex set $X \in \mathbb{R}^n$ described as the convex hull of k points $c_1, \ldots, c_k$. We would like to find an ellipsoid circumscribing these k points that has minimum volume, see Figure 6.

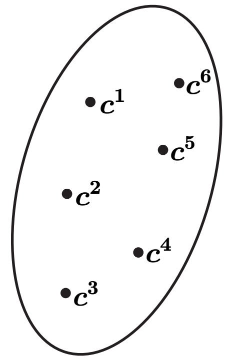

Figure 6: Illustration of the circumscribed ellipsoid problem.

Our problem can be written in the following form:

$$\begin{array}{llll} MCP: & \underset{R,z}{\text{minimize}} & \text{vol}\,(E_{R,z}) & \\ & \text{s.t.} & c_i \in E_{R,z}, & i = 1, \ldots, k, \end{array}$$

which is equivalent to:

$$\begin{array}{llll} MCP: & \underset{R,z}{\text{minimize}} & -\ln(\det(R)) & \\ & \text{s.t.} & (c_i - z)^T R (c_i - z) \leq 1, & i = 1, \ldots, k \\ & & R \succ 0, & \end{array}$$

Now factor $R = M^2$ where $M \succ 0$ (that is, M is a square root of R), and now MCP becomes:

$$\begin{array}{llll} MCP: & \underset{M,z}{\text{minimize}} & -\ln(\det(M^2)) & \\ & \text{s.t.} & (c_i - z)^T M^T M (c_i - z) \leq 1, & i = 1, \ldots, k, \\ & & M \succ 0. & \end{array}$$

Next notice the equivalence:

$$\begin{pmatrix} I & Mc_i - Mz \\ (Mc_i - Mz)^T & 1 \end{pmatrix} \succeq 0 \quad \Longleftrightarrow \quad (c_i - z)^T M^T M (c_i - z) \leq 1$$

In this way we can write MCP as:

$$\begin{array}{llll} MCP: & \underset{M,z}{\text{minimize}} & -2\ln(\det(M)) & \\ & \text{s.t.} & \begin{pmatrix} I & Mc_i - Mz \\ (Mc_i - Mz)^T & 1 \end{pmatrix} \succeq 0, & i = 1, \ldots, k, \\ & & M \succ 0. & \end{array}$$

Last of all, make the substitution $y = Mz$ to obtain:

$$\begin{array}{llll} MCP: & \underset{M,y}{\text{minimize}} & -2\ln(\det(M)) & \\ & \text{s.t.} & \begin{pmatrix} I & Mc_i - y \\ (Mc_i - y)^T & 1 \end{pmatrix} \succeq 0, & i = 1, \ldots, k, \\ & & M \succ 0. & \end{array}$$

Notice that this last program involves semidefinite constraints where all of the matrix coefficients are linear functions of the variables M and y. The objective function is the logarithmic barrier function $-\ln(\det(M))$. As discussed earlier, this function has the same remarkable properties as the logarithmic barrier function $-\sum_{j=1}^{n} \ln(x_j)$ does for linear optimization, and optimization of this function using Newton's method is extremely easy.

Finally, note that after solving the formulation of MCP above, we can recover the matrix R and the center z of the optimal ellipsoid by computing

$$R = M^2 \text{ and } z = M^{-1}y.$$

16.9 SDP in Control Theory

A variety of control and system problems can be cast and solved as instances of SDP. However, this topic is beyond the scope of these notes.

16.10 Interior-point Methods for SDP

The primal and dual SDP problems are:

$$\begin{array}{lll} SDP: & \text{minimize} & C \bullet X \\ & \text{s.t.} & A_i \bullet X = b_i, \ i = 1, \ldots, m, \\ & & X \succeq 0, \end{array}$$

and

$$\begin{array}{lll} SDD: & \text{maximize} & \sum\limits_{i=1}^{m} y_i b_i \\ & \text{s.t.} & \sum\limits_{i=1}^{m} y_i A_i + S = C \\ & & S \succeq 0. \end{array}$$

If X and (y, S) are feasible for the primal and the dual, the duality gap is:

$$C \bullet X - \sum_{i=1}^{m} y_i b_i = S \bullet X \geq 0.$$

Also,

$$S \bullet X = 0 \iff SX = 0.$$

Interior-point methods for semidefinite optimization are based on the logarithmic barrier function:

$$B(X) = -\sum_{j=1}^{n} \ln(\lambda_i(X)) = -\ln(\prod_{j=1}^{n} \lambda_i(X)) = -\ln(\det(X)).$$

Consider the logarithmic barrier problem $BSDP(\mu)$ parameterized by the positive barrier parameter μ:

$$\begin{array}{lll} BSDP(\mu): & \text{minimize} & C \bullet X - \mu \ln(\det(X)) \\ & \text{s.t.} & A_i \bullet X = b_i, \ i = 1, \ldots, m, \\ & & X \succ 0. \end{array}$$

Let $f_\mu(X)$ denote the objective function of $BSDP(\mu)$. Then it is not too difficult to derive:

$$-\nabla f_\mu(X) = C - \mu X^{-1},$$

and so the Karush-Kuhn-Tucker conditions for $BSDP(\mu)$ are:

$$\begin{cases} A_i \bullet X = b_i, \ i = 1, \ldots, m, \\ X \succ 0, \\ C - \mu X^{-1} = \sum\limits_{i=1}^{m} y_i A_i. \end{cases}$$

We can define

$$S = \mu X^{-1},$$

which implies

$$XS = \mu I,$$

and we can rewrite the Karush-Kuhn-Tucker conditions as:

$$\begin{cases} A_i \bullet X = b_i, \ i = 1, \ldots, m, \\ X \succ 0 \\ \sum\limits_{i=1}^{m} y_i A_i + S = C \\ XS = \mu I. \end{cases}$$

It follows that if (X, y, S) is a solution of this system, then X is feasible for SDP, (y, S) is feasible for SDD, and the resulting duality gap is

$$S \bullet X = \sum_{i=1}^{n} \sum_{j=1}^{n} S_{ij} X_{ij} = \sum_{j=1}^{n} (SX)_{jj} = \sum_{j=1}^{n} (\mu I)_{jj} = n\mu.$$

This suggests that we try solving $BSDP(\mu)$ for a variety of values of μ as $\mu \to 0$.

Interior-point methods for SDP are very similar to those for linear optimization, in that they use Newton's method to solve the KKT system as $\mu \to 0$.

16.11 Website for SDP

A good website for semidefinite programming is:

`http://www-user.tu-chemnitz.de/ helmberg/semidef.html`.