Conic Programming

Theory and Algorithms

Sirong Luo

Faculty of Statistics and Data Science
Shanghai University of Finance and Economics

Table of Contents

1. Conic Programming
2. Duality and Optimality Conditions
3. Algorithms

Table of Contents

1 Conic Programming

2 Duality and Optimality Conditions

3 Algorithms

Conic Programming

A conic program in standard form is an optimization problem of the form

$$\begin{aligned} \min_{\mathbf{x}} \ & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} \ & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{D}\mathbf{x} - \mathbf{d} \in \mathcal{K} \end{aligned} \tag{18.1}$$

for some vectors and matrices
$\mathbf{c} \in \mathbb{R}^n, \mathbf{b} \in \mathbb{R}^m, \mathbf{d} \in \mathbb{R}^p, \mathbf{A} \in \mathbb{R}^{m \times n}, \mathbf{D} \in \mathbb{R}^{p \times n}$ and some closed convex cone $\mathcal{K} \subseteq \mathbb{R}^p$.
When $\mathcal{K} = \mathbb{R}^p_+$ the problem (18.1) is a linear program. However, conic programming is far more general. We next discuss two particularly important classes of conic programs, namely second-order and semidefinite programming.

Second-Order Programming

Second-Order Programming

The second-order cone, also known as the Lorenz cone or the ice-cream cone, is defined as follows:

$$\mathbb{L}_n = \left\{ \mathbf{x} = \begin{bmatrix} x_0 \\ \overline{\mathbf{x}} \end{bmatrix} \in \mathbb{R}^n : \|\overline{\mathbf{x}}\|_2 \leq x_0 \right\}$$

See Figure 18.1.

Figure

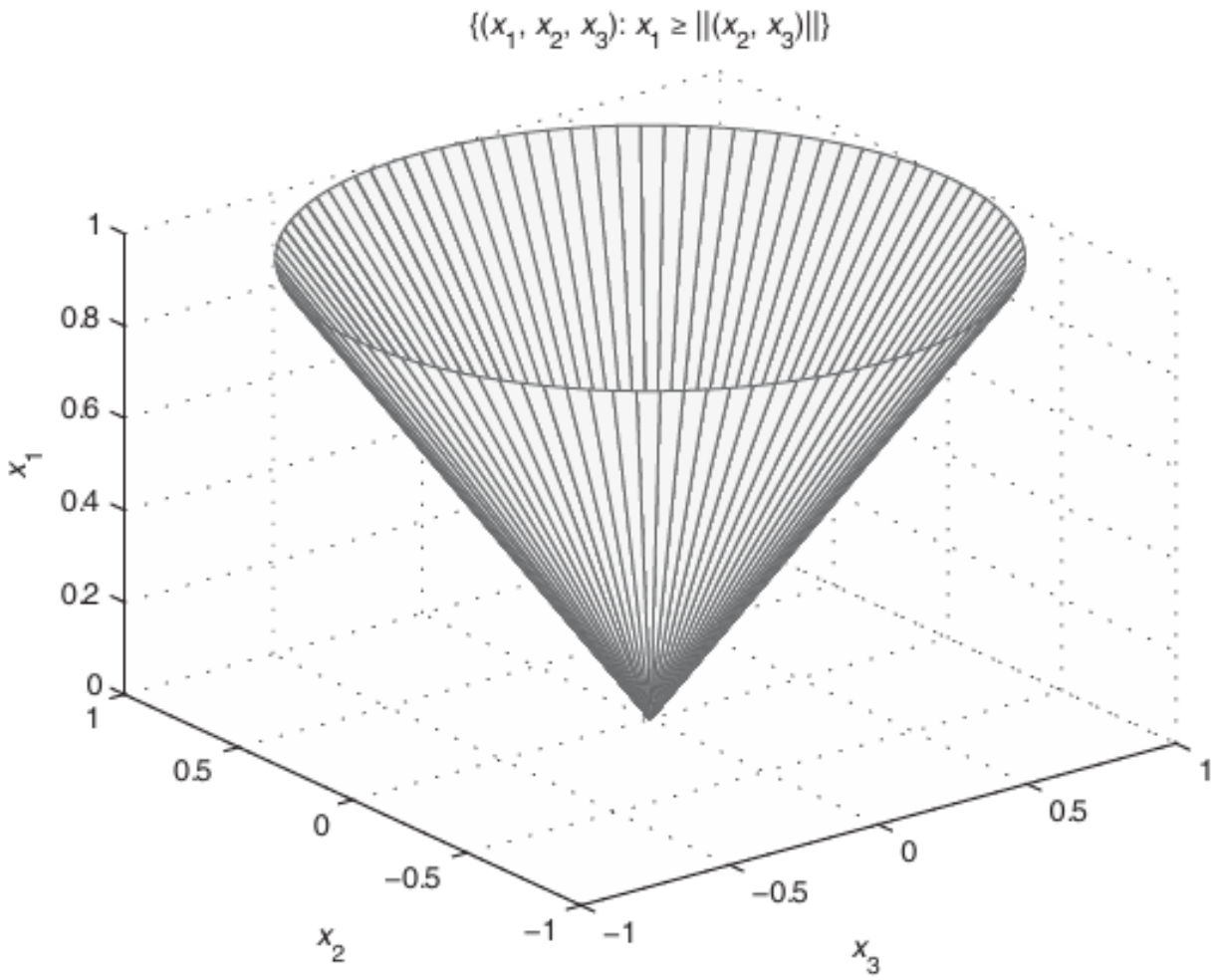

Figure: Second-order cone

Second-Order Programming

A second-order program is a problem of the form (18.1) where $\mathcal{K}$ is a direct product of second-order cones; that is,

$$\mathcal{K} = \mathbb{L}_{n_1} \times \cdots \times \mathbb{L}_{n_r}$$

for some positive integers $n_1, \ldots, n_r$. We next illustrate the modeling power of second-order programming by showing that a convex quadratically constrained quadratic program can be recast as a second-order program. In particular, second-order programming generalizes both linear programming and convex quadratic programming.

Consider a convex quadratically constrained quadratic program of the form

$$\begin{aligned} \min_{\mathbf{x}} & \mathbf{c}_0^\top \mathbf{x} + \frac{1}{2}\mathbf{x}^\top \mathbf{Q}_0 \mathbf{x} \\ \text{s.t. } & \mathbf{c}_i^\top \mathbf{x} + \frac{1}{2}\mathbf{x}^\top \mathbf{Q}_i \mathbf{x} \le b_i, i = 1, \ldots, m \end{aligned} \tag{18.2}$$

where $\mathbf{c}_i \in \mathbb{R}^n, \mathbf{Q}_i \in \mathbb{S}_+^n$ for $i = 0, 1, \ldots, m$ and $b_i \in \mathbb{R}$ for $i = 1, \ldots, m$. Here $\mathbb{S}_+^n$ denotes the family of $n \times n$ positive semidefinite matrices. Observe that (18.2) can be rewritten as

$$\begin{aligned} \min_{\mathbf{x}, t} & t \\ \text{s.t. } & \mathbf{c}_0^\top \mathbf{x} + \frac{1}{2}\mathbf{x}^\top \mathbf{Q}_0 \mathbf{x} \le t \\ & \mathbf{c}_i^\top \mathbf{x} + \frac{1}{2}\mathbf{x}^\top \mathbf{Q}_i \mathbf{x} \le b_i, i = 1, \ldots, m \end{aligned} \tag{18.3}$$

The following step is key in the formulation of (18.2) as a second-order program: given $\mathbf{Q} \in \mathbb{S}^n_+, \mathbf{c} \in \mathbb{R}^n, b \in \mathbb{R}$ the quadratic inequality

$$\mathbf{c}^\top \mathbf{x} + \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} \leq b$$

can be formulated as a second-order cone constraint. To see that, observe that because $\mathbf{Q} \in \mathbb{S}^n_+$ there exists $\mathbf{L} \in \mathbb{R}^{n \times p}$ such that $\mathbf{Q} = \mathbf{L}\mathbf{L}^\top$ (in particular the Cholesky factorization satisfies this requirement). Therefore

$$\mathbf{c}^\top \mathbf{x} + \frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} \leq b \quad \Leftrightarrow \quad \mathbf{c}^\top \mathbf{x} + \frac{1}{2}\left\|\mathbf{L}^\top \mathbf{x}\right\|^2 \leq b \quad \Leftrightarrow \quad \begin{bmatrix} b - \mathbf{c}^\top \mathbf{x} + 1 \\ b - \mathbf{c}^\top \mathbf{x} - 1 \\ \sqrt{2}\mathbf{L}^\top \mathbf{x} \end{bmatrix} \in \mathbb{L}$$

It thus follows that (18.3) and in turn (18.2) can be rewritten as the following second-order program:

$$
\begin{aligned}
\min_{\mathbf{x},t} \quad & t \\
\text{s.t.} \quad & \begin{bmatrix} t - \mathbf{c}_0^\top \mathbf{x} + 1 \\ t - \mathbf{c}_0^\top \mathbf{x} - 1 \\ \sqrt{2}\mathbf{L}_0^\top \mathbf{x} \end{bmatrix} \in \mathbb{L}_{p_0+2} \\
& \begin{bmatrix} b_i - \mathbf{c}_i^\top \mathbf{x} + 1 \\ b_i - \mathbf{c}_i^\top \mathbf{x} - 1 \\ \sqrt{2}\mathbf{L}_i^\top \mathbf{x} \end{bmatrix} \in \mathbb{L}_{p_i+2}, i = 1, \ldots, m
\end{aligned}
$$

where $\mathbf{L}_i \in \mathbb{R}^{n \times p_i}$ such that $\mathbf{Q}_i = \mathbf{L}_i \mathbf{L}_i^\top$ for $i = 0, 1, \ldots, m$.

Application

Tracking Error and Volatility Constraints

In the context of quantitative asset management, portfolios are typically chosen relative to some predetermined benchmark, as we discussed in Section 6.5. As a consequence, it is common to use a constraint on the active risk (also known as tracking error) instead of, or in addition to, the total risk. More precisely, suppose $\mathbf{x}$ denotes the vector of percentage holdings in a portfolio. Let $\mathbf{r}$ and r_B denote respectively the vector of asset returns and the benchmark return. Recall that the active return is the difference $\mathbf{r}^\top \mathbf{x} - r_B$ between the portfolio return and the benchmark return. The active risk is the variance of the active return. If $\mathbf{x}^B$ denotes the vector of percentage holdings in the benchmark, then the active risk can be written as

$$\text{var}\left(\mathbf{r}^\top\left(\mathbf{x}-\mathbf{x}^B\right)\right) = \left(\mathbf{x}-\mathbf{x}^B\right)^\top \mathbf{V}\left(\mathbf{x}-\mathbf{x}^B\right)$$

where $\mathbf{V}$ is the covariance matrix of asset returns.

A typical mean-variance model for benchmark-relative portfolio management has the following form:

$$\begin{aligned} \max_{\mathbf{x}} \; & \boldsymbol{\alpha}^\top \mathbf{x} \\ \text{s.t.} \; & \left(\mathbf{x} - \mathbf{x}^B\right)^\top \mathbf{V} \left(\mathbf{x} - \mathbf{x}^B\right) \leq \bar{\psi}^2 \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{C}\mathbf{x} \leq \mathbf{d}. \end{aligned} \tag{18.4}$$

Note that this is not a quadratic program because it has a nonlinear constraint. However, the problem (18.4) is a convex quadratically constrained quadratic program of the form (18.2) discussed in Section 18.1.1. Therefore, it has a straightforward formulation as a second-order conic program.

The above model can be readily extended to include multiple measures of risk. For instance, the following model, which is an extension of the model discussed by Jorion (2003) that enforces upper bound constraints on both total risk and tracking error, also has a straightforward second-order conic programming formulation:

$$
\begin{aligned}
\max_{\mathbf{x}} \; & \boldsymbol{\alpha}^\top \mathbf{x} \\
\text{s.t.} \; & \left(\mathbf{x} - \mathbf{x}^B\right)^\top \mathbf{V} \left(\mathbf{x} - \mathbf{x}^B\right) \leq \bar{\psi}^2 \\
& \mathbf{x}^\top \mathbf{V} \mathbf{x} \leq \bar{\sigma}^2 \\
& \mathbf{A}\mathbf{x} = \mathbf{b} \\
& \mathbf{C}\mathbf{x} \leq \mathbf{d}.
\end{aligned} \tag{18.5}
$$

Semidefinite Programming

Some applications, such as the approximation of covariance matrices discussed below, lead to conic optimization models involving the space of symmetric matrices and the cone of positive semidefinite matrices described next. Let $\mathbb{S}^n$ denote the space of $n \times n$ symmetric matrices. Although this space is equivalent to $\mathbb{R}^{n(n+1)/2}$, it is more convenient and customary to treat it as a space of matrices. A matrix $\mathbf{X} \in \mathbb{S}^n$ is positive semidefinite if

$$\mathbf{u}^\top \mathbf{X}\mathbf{u} \geq 0 \text{ for all } \mathbf{u} \in \mathbb{R}^n$$

It is a common convention to write $\mathbf{X} \succeq \mathbf{0}$ to indicate that $\mathbf{X} \in \mathbb{S}^n$ is positive semidefinite. The cone of positive semidefinite matrices $\mathbb{S}^n_+$ is defined as

$$\mathbb{S}^n_+ := \{\mathbf{X} \in \mathbb{S}^n : \mathbf{X} \succeq \mathbf{0}\}$$

See Figure 18.2.

Figure

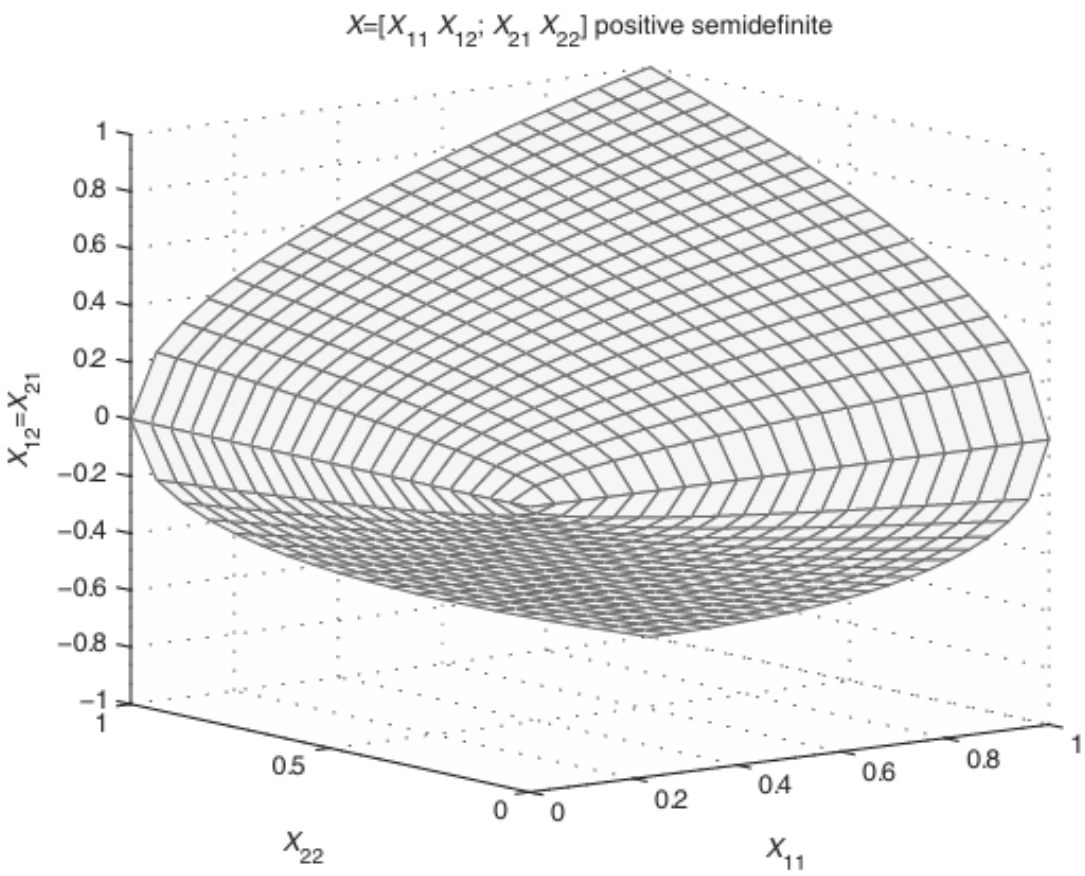

Figure: Second-order cone

A semidefinite program is a problem of the form (18.1) where $\mathcal{K}$ is the cone of positive semidefinite matrices.Endow the space $\mathbb{S}^n$ of symmetric $n \times n$ matrices with the following Frobenius inner product. For $\mathbf{X}, \mathbf{S} \in \mathbb{S}^n$ let

$$\mathbf{X} \bullet \mathbf{S} := \text{trace}(\mathbf{XS}) = \sum_{i,j} X_{ij} S_{ij}$$

A semidefinite program is in standard form if it is written as

$$\begin{array}{ll} \min_{\mathbf{X}} & \mathbf{C} \bullet \mathbf{X} \\ \text{s.t.} & \mathbf{AX} = \mathbf{b} \\ & \mathbf{X} \succeq \mathbf{0}, \end{array}$$

where $\mathbf{C} \in \mathbb{S}^n, \mathbf{b} \in \mathbb{R}^m$, and $\mathbf{A} : \mathbb{S}^n \to \mathbb{R}^m$ is a linear mapping.

Application

Approximating Covariance Matrices We next illustrate the modeling power of semidefinite programming by showing that a covariance estimation problem can be recast as a semidefinite program. To that end, recall that any proper covariance matrix must be symmetric and positive semidefinite. Suppose $\hat{\mathbf{V}} \in \mathbb{S}^n$ is an estimate of a covariance matrix that is not necessarily positive semidefinite and consider the problem of finding the positive semidefinite matrix that is closest to $\hat{\mathbf{V}}$; that is,

$$\begin{aligned} \min_{\mathbf{X}} & \|\mathbf{X} - \hat{\mathbf{V}}\| \\ \text{s.t.} & \ \mathbf{X} \succeq \mathbf{0}. \end{aligned} \tag{18.6}$$

This problem can be formulated as

$$\begin{aligned} \min_{\mathbf{X},t} & \quad t \\ \text{s.t.} & \quad \|\mathbf{X} - \hat{\mathbf{V}}\| \leq t \\ & \quad \mathbf{X} \succeq \mathbf{0}. \end{aligned}$$

If the norm $\|\mathbf{X} - \hat{\mathbf{V}}\|$ is the Frobenius norm

$$\|\mathbf{X} - \hat{\mathbf{V}}\|_F := \sqrt{\sum_{i,j} \left(X_{ij} - \hat{V}_{ij}\right)^2}$$

then the constraint $\|\mathbf{X} - \hat{\mathbf{V}}\|_F \leq t$ can be written as a second-order cone constraint and it follows that the above covariance estimation problem (18.6) can be written as a conic program over the Cartesian product of a second-order cone and a semidefinite cone. As we detail in the exercises at the end of this chapter, problem (18.6) can also be formulated as a conic program over a suitable semidefinite cone for other choices of norms such as the operator norm or the infinity norm.

Table of Contents

1 Conic Programming

2 Duality and Optimality Conditions

3 Algorithms

Duality and Optimality Conditions

As in linear and quadratic programming, there is a dual conic program associated with every primal conic program. The construction of the dual conic program relies on the following more fundamental construction. Let $\mathcal{K} \subseteq \mathbb{R}^p$ be a closed convex cone. The dual cone $\mathcal{K}^* \subseteq \mathbb{R}^p$ of $\mathcal{K}$ is defined as

$$\mathcal{K}^* := \left\{ \mathbf{s} \in \mathbb{R}^p : \mathbf{s}^\top \mathbf{x} \geq 0 \text{ for all } \mathbf{x} \in \mathcal{K} \right\}.$$

It is easy to see that $\mathcal{K}^* \subseteq \mathbb{R}^p$ is also a closed convex cone. Just as we did for linear and quadratic programming, the dual problem can be derived via the following Lagrangian function associated with (18.1):

$$L(\mathbf{x}, \mathbf{y}, \mathbf{s}) := \mathbf{c}^\top \mathbf{x} + \mathbf{y}^\top (\mathbf{b} - \mathbf{A}\mathbf{x}) + \mathbf{s}^\top (\mathbf{d} - \mathbf{D}\mathbf{x}).$$

The constraints of (18.1) can be encoded via the Lagrangian function. For a given vector $\mathbf{x}$

$$\max_{\substack{\mathbf{y},\mathfrak{x}^* \\ \mathbf{s}\in\mathcal{K}^*}} L(\mathbf{x},\mathbf{y},\mathbf{s}) = \begin{cases} \mathbf{c}^\top\mathbf{x} & \text{if } \mathbf{A}\mathbf{x}=\mathbf{b} \text{ and } \mathbf{D}\mathbf{x}-\mathbf{d}\in\mathcal{K} \\ +\infty & \text{otherwise.} \end{cases}$$

Therefore the primal problem (18.1) can be written as

$$\min_{\mathbf{x}} \max_{\substack{\mathbf{y},\mathbf{s} \\ \mathbf{s}\in\mathcal{K}^*}} L(\mathbf{x},\mathbf{y},\mathbf{s})$$

The dual problem is obtained by flipping the order of the min and max operations:

$$\max_{\substack{\mathbf{y},* \\ \mathbf{s}\in\mathcal{K}^*}} \min_{\mathbf{x}} L(\mathbf{x},\mathbf{y},\mathbf{s})$$

It is easy to see that the dual problem can be written as follows:

$$\begin{aligned} \max_{\mathbf{y},\mathbf{s}} \quad & \mathbf{b}^\top \mathbf{y} + \mathbf{d}^\top \mathbf{s} \\ \text{s.t.} \quad & \mathbf{A}^\top \mathbf{y} + \mathbf{D}^\top \mathbf{s} = \mathbf{c} \\ & \mathbf{s} \in \mathcal{K}^*. \end{aligned}$$

In particular, when the primal problem is in the following standard form,

$$\begin{array}{ll} \min_{\mathbf{x}} & \mathbf{c}^\top \mathbf{x} \\ \text{s.t.} & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \in \mathcal{K} \end{array}$$

the dual problem is

$$\begin{array}{ll} \max_{\mathbf{y},\mathbf{s}} & \mathbf{b}^\top \mathbf{y} \\ \text{s.t.} & \mathbf{A}^\top \mathbf{y} + \mathbf{s} = \mathbf{c} \\ & \mathbf{s} \in \mathcal{K}^* \end{array}$$

Observe that the dual problem of a conic program is again a conic program. As in linear and quadratic programming, there is a deep connection between the primal problem (18.1) and its dual (18.7). The following result follows by construction.

Theorem

Theorem 18.1 (Weak duality) Assume $\mathbf{x}$ is a feasible point for (18.1) and $(\mathbf{y}, \mathbf{s})$ is a feasible point for (18.7). Then

$$\mathbf{b}^\top \mathbf{y} + \mathbf{d}^\top \mathbf{s} \leq \mathbf{c}^\top \mathbf{x}$$

Proof If $\mathbf{x}$ and $(\mathbf{y}, \mathbf{s})$ satisfy the above assumptions then

$$\begin{aligned} \mathbf{b}^\top \mathbf{y} + \mathbf{d}^\top \mathbf{s} &\leq (\mathbf{A}\mathbf{x})^\top \mathbf{y} + (\mathbf{D}\mathbf{x})^\top \mathbf{s} \\ &= \left(\mathbf{A}^\top \mathbf{y} + \mathbf{D}^\top \mathbf{s}\right)^\top \mathbf{x} \\ &= \mathbf{c}^\top \mathbf{x}. \end{aligned}$$

Theorem 18.2 (Strong duality) Suppose the problems (18.1) and (18.7) satisfy the Slater condition. Then both problems have optimal solutions and their optimal values are the same.

Theorem 18.3 (Optimality conditions) The vectors $\mathbf{x} \in \mathbb{R}^n$ and $(\mathbf{y}, \mathbf{s}) \in \mathbb{R}^m \times \mathbb{R}^n$ are optimal solutions to (18.1) and (18.7) respectively if

$$\begin{aligned} \mathbf{c} - \mathbf{A}^\top \mathbf{y} - \mathbf{D}^\top \mathbf{s} &= \mathbf{0} \\ \mathbf{A}\mathbf{x} - \mathbf{b} &= \mathbf{0} \\ \mathbf{D}\mathbf{x} - \mathbf{d} &\in \mathcal{K} \\ \mathbf{s} &\in \mathcal{K}^* \\ \mathbf{s}^\top (\mathbf{D}\mathbf{x} - \mathbf{d}) &= 0 \end{aligned} \tag{18.8}$$

The following partial converse also holds: if (18.1) and (18.7) satisfy the Slater condition then they both have optimal solutions $\mathbf{x}$ and ($\mathbf{y}, \mathbf{s}$) that satisfy (18.8).

For a conic program in standard form, the optimality conditions (18.8) can be written as follows:

$$\begin{aligned} \mathbf{A}^\top \mathbf{y} + \mathbf{s} &= \mathbf{c} \\ \mathbf{A}\mathbf{x} &= \mathbf{b} \\ \mathbf{x} &\in \mathcal{K} \\ \mathbf{s} &\in \mathcal{K}^* \\ \mathbf{s}^\top \mathbf{x} &= 0 \end{aligned} \tag{18.9}$$

Table of Contents

1 Conic Programming

2 Duality and Optimality Conditions

3 Algorithms

Interior-point Methods

By relying on the structure of the cone $\mathcal{K}$, the main algorithmic template of interior-point methods, such as the one described in Chapter 2 and in Chapter 5 , can be extended to a larger class of conic programs. The central idea is to generate a sequence of points that converges to a solution to (18.9). We next sketch the gist of interior-point methods for semidefinite programming.

For convenience of exposition, we consider a semidefinite program in standard form:

$$\begin{aligned} \min_{\mathbf{X}} \ & \mathbf{C} \bullet \mathbf{X} \\ \text{s.t.} \ & \mathbf{A}\mathbf{X} = \mathbf{b} \\ & \mathbf{X} \succeq \mathbf{0}, \end{aligned} \tag{18.10}$$

where $\mathbf{C} \in \mathbb{S}^n, \mathbf{b} \in \mathbb{R}^m$, and $\mathbf{A} : \mathbb{S}^n \to \mathbb{R}^m$ is a linear mapping. As the exercises at the end of this chapter detail, the dual of (18.10) is the semidefinite program

$$\begin{aligned} \min_{\mathbf{y},\mathbf{S}} \ & \mathbf{b}^\top \mathbf{y} \\ \text{s.t.} \ & \mathbf{A}^* \mathbf{y} + \mathbf{S} = \mathbf{C} \\ & \mathbf{S} \succeq \mathbf{0}, \end{aligned} \tag{18.11}$$

where $\mathbf{A}^* : \mathbb{R}^m \to \mathbb{S}^n$ denotes the adjoint of $\mathbf{A}$; that is, the unique linear mapping satisfying

We will rely on the following key property of positive semidefinite matrices:

$$\mathbf{X}, \mathbf{S} \in \mathbb{S}^n_+ \text{ and } \mathbf{X} \bullet \mathbf{S} = 0 \quad \Rightarrow \quad \mathbf{XS} = \mathbf{0} \tag{18.12}$$

From Theorem 18.3 and (18.12) it follows that $\mathbf{X}$ and $(\mathbf{y}, \mathbf{S})$ are optimal solutions to (18.10) and (18.11) respectively if

$$\begin{aligned} \mathbf{A}^*\mathbf{y} + \mathbf{S} &= \mathbf{C} \\ \mathbf{AX} &= \mathbf{b} \\ \mathbf{XS} &= \mathbf{0} \\ \mathbf{X}, \mathbf{S} &\succeq \mathbf{0}. \end{aligned} \tag{18.13}$$

As in the linear programming case, interior-point methods for semidefinite programming generate a sequence of iterates that satisfy $\mathbf{X}, \mathbf{S} \succ \mathbf{0}$. Each iteration of the algorithm aims to make progress towards satisfying $\mathbf{A}^*\mathbf{y} + \mathbf{S} = \mathbf{C}$, $\mathbf{AX} = \mathbf{b}$, and $\mathbf{XS} = \mathbf{0}$.

Given $\mu > 0$, let $(\mathbf{X}(\mu), \mathbf{y}(\mu), \mathbf{S}(\mu))$ be the solution to the following perturbed version of the above optimality conditions:

$$\begin{bmatrix} \mathbf{A}^*\mathbf{y} + \mathbf{S} - \mathbf{C} \\ \mathbf{AX} - \mathbf{b} \\ \mathbf{XS} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mu\mathbf{I} \end{bmatrix}, \mathbf{X}, \mathbf{S} \succ \mathbf{0}$$

The first condition above can be written as $\mathbf{r}_\mu(\mathbf{X}, \mathbf{y}, \mathbf{S}) = \mathbf{0}$ for the residual vector:

$$\mathbf{r}_\mu(\mathbf{X}, \mathbf{y}, \mathbf{S}) := \begin{bmatrix} \mathbf{A}^*\mathbf{y} + \mathbf{S} - \mathbf{C} \\ \mathbf{A}\mathbf{X} - \mathbf{b} \\ \mathbf{X}\mathbf{S} - \mu\mathbf{I} \end{bmatrix}$$

The central path is the set $\{(\mathbf{X}(\mu), \mathbf{y}(\mu), \mathbf{S}(\mu)) : \mu > 0\}$. It is intuitively clear that $(\mathbf{X}(\mu), \mathbf{y}(\mu), \mathbf{S}(\mu))$ converges to an optimal solution to both (18.10) and (18.11). This suggests the following algorithmic strategy: suppose $(\mathbf{X}, \mathbf{y}, \mathbf{S})$ is "near" $(\mathbf{X}(\mu), \mathbf{y}(\mu), \mathbf{S}(\mu))$ for some $\mu > 0$. Use $(\mathbf{X}, \mathbf{y}, \mathbf{S})$ to move to a better point $(\mathbf{X}^+, \mathbf{y}^+, \mathbf{S}^+)$"near" $(\mathbf{X}(\mu^+), \mathbf{y}(\mu^+), \mathbf{S}(\mu^+))$for some $\mu^+ < \mu$.

It can be shown that if a point $(\mathbf{X}, \mathbf{y}, \mathbf{S})$ is on the central path, then the corresponding value of μ satisfies $\mathbf{X} \bullet \mathbf{S} = n\mu$. Likewise, given $\mathbf{X}, \mathbf{S} \succ \mathbf{0}$, define

$$\mu(\mathbf{X}, \mathbf{S}) := \frac{\mathbf{X} \bullet \mathbf{S}}{n}.$$

To move from a current point $(\mathbf{X}, \mathbf{y}, \mathbf{S})$ to a new point, we use a suitable Newton step; that is, the solution to the following system of equations obtained as a linearization of the system of nonlinear equations (18.14) $\mathbf{r}_\mu(\mathbf{X}, \mathbf{y}, \mathbf{S}) = \mathbf{0}$:

$$\begin{bmatrix} \mathbf{0} & \mathbf{A}^* & \mathbf{I} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{F} & \mathbf{0} & \mathbf{G} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{X} \\ \Delta\mathbf{y} \\ \Delta\mathbf{S} \end{bmatrix} = \begin{bmatrix} \mathbf{C} - \mathbf{A}^\top\mathbf{y} - \mathbf{S} \\ \mathbf{b} - \mathbf{A}\mathbf{X} \\ \mu\mathbf{X}^{-1} - \mathbf{S} \end{bmatrix}$$

for some suitably chosen mappings $\mathbf{F}, \mathbf{G}$ that depend on the current $\mathbf{X}, \mathbf{S}$. The details of these mappings are somewhat technical and related to nuances concerning the space of symmetric $n \times n$ matrices. Further details can be found in Renegar (2001) and in the exercises at the end of the chapter.

Algorithm

Algorithm 18.1 presents a template for an interior-point method for semidefinite programming.

Algorithm

Algorithm 18.1 presents a template for an interior-point method for semidefinite programming.

Algorithm 18.1 Interior-point method for semidefinite programming

1: choose $\mathbf{X}^0, \mathbf{S}^0 \succ 0$

2: for $k = 0, 1, \ldots$ do

3: solve the Newton system (18.14) for $(\mathbf{X}, \mathbf{y}, \mathbf{S}) = \left(\mathbf{X}^k, \mathbf{y}^k, \mathbf{S}^k\right)$ and $\mu := 0.1\mu\left(\mathbf{X}^k, \mathbf{S}^k\right)$

4: choose a step length $\alpha \in (0, 1]$ and set $\left(\mathbf{X}^{k+1}, \mathbf{y}^{k+1}, \mathbf{S}^{k+1}\right) = \left(\mathbf{X}^k, \mathbf{y}^k, \mathbf{S}^k\right) + \alpha(\Delta\mathbf{X}, \Delta\mathbf{y}, \Delta\mathbf{S})$

5: end for

The step length α in step 4 should be chosen so that $\mathbf{X}^{k+1}, \mathbf{S}^{k+1} \succ \mathbf{0}$ and the size of $\mathbf{r}_\mu\left(\mathbf{X}^{k+1}, \mathbf{y}^{k+1}, \mathbf{S}^{k+1}\right)$ is sufficiently smaller than $\mathbf{r}_\mu\left(\mathbf{X}^{k}, \mathbf{y}^{k}, \mathbf{S}^{k}\right)$. A line-search procedure such as the one described in Algorithm 2.4 in Chapter 2 can be used for choosing the step length α.