Chapter VI: Conic Programming

1 Conic Programming

In this section, we introduce an important class of convex optimization problems, which generalizes the class of linear programs in a natural fashion.

Definition 1 (Conic Programming)**.** Let K be a proper cone. A *conic program in standard form* (relative to the cone K) is an optimization problem of the form

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\textbf{minimize}} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ s.t. \quad & A\boldsymbol{x} = \boldsymbol{b}, \\ & \boldsymbol{x} \succeq_K \boldsymbol{0}. \end{aligned} \tag{$\mathrm{P}_{\mathrm{Cone}}$}$$

- When the cone K is the nonnegative orthant $\mathbb{R}^n_+$, we say that Problem ($\mathrm{P}_{\mathrm{Cone}}$) is a *Linear Program* (LP).
- When the cone K is a direct product of Lorrentz cones, we say that ($\mathrm{P}_{\mathrm{Cone}}$) is a *Second Order Cone Program* (SOCP). Some authors also use the term *conic quadratic program* (CQP). For consistency, we will adopt the following notation for the Lorrentz cone:

$$\mathbb{L}^n_+ := \{\boldsymbol{x} \in \mathbb{R}^n : \ \sqrt{x_1^2 + \ldots + x_{n-1}^2} \leq x_n\}.$$

- When the cone K is the semidefinite cone $\mathbb{S}^n_+$, we say that ($\mathrm{P}_{\mathrm{Cone}}$) is a *Semidefinite Program* (SDP).

More generally, we call LP/SOCP/SDP any optimization problem that is "trivially equivalent" to a conic problem in the standard form ($\mathrm{P}_{\mathrm{Cone}}$). We define this more formally next.

Definition 2. Let $\boldsymbol{x} \in \mathbb{R}^n$ be a decision variable. We say that a constraint of the form

$$\|A\boldsymbol{x} - \boldsymbol{b}\| \leq \boldsymbol{c}^T\boldsymbol{x} + d,$$

is a *second order cone inequality*.

Let $F : \mathbb{R}^n \to \mathbb{S}^m$ be an affine function. We say that a constraint of the form "$F(\boldsymbol{x}) \succeq 0$," where the inequality $\succeq$ is relative to $\mathbb{S}^m_+$, is a *linear matrix inequality*.

In brief, the general idea to remember is that

- any optimization problem that contains only linear equalities and inequalities is an LP;
- if in addition it contains some *second order cone inequalities*, it is an SOCP;
- finally, if in addition, is also contains *linear matrix inequalities*, then it is an SDP (we will understand later why a program with second order cone inequalities can be seen as an SDP...).

Definition 3. Let $K_i \subset \mathcal{V}_i$ be a proper cone, where $\mathcal{V}_i$ is a vector space (of finite dimension), and let $F_i : \mathbb{R}^n \to \mathcal{V}_i$ be an affine mapping ($\forall i \in [q]$). Any optimization problem of the form

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\textbf{minimize}} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ s.t. \quad & A\boldsymbol{x} = \boldsymbol{b}, \\ & F_i(\boldsymbol{x}) \succeq_{K_i} \boldsymbol{0}, \qquad (\forall i \in [q]). \end{aligned}$$

is equivalent to a conic program in standard form ($\mathrm{P}_{\mathrm{Cone}}$), and is hence called a *conic program*.

Note that the above definition is written with abstract vector spaces $\mathcal{V}_i$ (rather than, e.g., $\mathbb{R}^{n_i}$). This allows to handle the case where $F_i(\boldsymbol{x})$ is a matrix (for semidefinite programming). We will now show in several examples how the reduction to the standard form works.

Example:
The following optimization problem is an LP:

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\textbf{minimize}} \quad & \boldsymbol{c}_0^T\boldsymbol{x} \\ s.t. \quad & F\boldsymbol{x} = \boldsymbol{f}, \\ & H\boldsymbol{x} \geq \boldsymbol{h}. \end{aligned}$$

To convert it to the standard form ($\mathrm{P}_{\mathrm{Cone}}$), we proceed in two steps. Denote by m_1 and m_2 the dimensions of $\boldsymbol{f}$ and $\boldsymbol{h}$, respectively. First, we introduce a slack variable $\boldsymbol{z} = H\boldsymbol{x} - \boldsymbol{h} \in \mathbb{R}^{m_2}$. This yields the equivalent formulation:

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n,\ \boldsymbol{z}\in\mathbb{R}^{m_2}}{\textbf{minimize}} \quad & \boldsymbol{c}_0^T\boldsymbol{x} \\ s.t. \quad & F\boldsymbol{x} = \boldsymbol{f}, \\ & H\boldsymbol{x} - \boldsymbol{z} = \boldsymbol{h}. \\ & \boldsymbol{z} \geq \boldsymbol{0}. \end{aligned}$$

Now, it remains to deal with the fact that the variable $\boldsymbol{x}$ is *unconstrained*, while the standard form ($\mathrm{P}_{\mathrm{Cone}}$) only allows for variables in the cone K. So, the idea is to replace $\boldsymbol{x}$ by the difference between two vectors of nonnegative variables: #1

$$\boldsymbol{x} \in \mathbb{R}^n \iff \exists \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^n_+ : \boldsymbol{x} = \boldsymbol{x}_1 - \boldsymbol{x}_2.$$

We thus obtain the following problem:

$$\begin{aligned} \underset{\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathbb{R}^n,\ \boldsymbol{z}\in\mathbb{R}^{m_2}}{\textbf{minimize}} \quad & \boldsymbol{c}_0^T(\boldsymbol{x}_1 - \boldsymbol{x}_2) \\ s.t. \quad & F(\boldsymbol{x}_1 - \boldsymbol{x}_2) = \boldsymbol{f}, \\ & H(\boldsymbol{x}_1 - \boldsymbol{x}_2) - \boldsymbol{z} = \boldsymbol{h}. \\ & \boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{z} \geq \boldsymbol{0} \end{aligned}$$

Finally, we obtain a problem of the standard form, by setting

$$\boldsymbol{c}^T = [\boldsymbol{c}_0^T, -\boldsymbol{c}_0^T, \boldsymbol{0}^T]^T \in \mathbb{R}^{2n+m_2}, \qquad A = \begin{bmatrix} F & -F & O \\ H & -H & -I \end{bmatrix} \in \mathbb{R}^{(m_1+m_2)\times(2n+m_2)}, \qquad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{h} \end{bmatrix} \in \mathbb{R}^{m_1+m_2},$$

and the cone K is the nonnegative orthant $\mathbb{R}^{2n+m_2}_+$.

Example:
The following problem is an SOCP:

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\textbf{minimize}} \quad & \boldsymbol{c}_0^T\boldsymbol{x} \\ s.t. \quad & F\boldsymbol{x} = \boldsymbol{f}, \\ & H\boldsymbol{x} \geq \boldsymbol{g}, \\ & \|A_i\boldsymbol{x} + \boldsymbol{b}_i\| \leq \boldsymbol{c}_i^T\boldsymbol{x} + d_i, \quad (\forall i \in [m]). \end{aligned} \tag{1}$$

The fact that the variable $\boldsymbol{x}$ is unconstrained can be handled exactly as in the previous example. Now, we need to rewrite the linear inequalities and second order cone inequalities as in the standard form ($\mathrm{P}_{\mathrm{Cone}}$). If the rows of H are $\boldsymbol{h}_1^T, \dots, \boldsymbol{h}_p^T$, then we introduce the slack variables

$$\boldsymbol{y}_i = \begin{bmatrix} 0 \\ \boldsymbol{h}_i^T\boldsymbol{x} - g_i \end{bmatrix} \in \mathbb{R}^2,$$

#2

and for all $i \in [m]$, we introduce the slack variable

$$\boldsymbol{z}_i = \begin{bmatrix} A_i\boldsymbol{x} + \boldsymbol{b}_i \\ \boldsymbol{c}_i^T\boldsymbol{x} + d_i \end{bmatrix} \in \mathbb{R}^{n_i+1},$$

where n_i is the dimension of $\boldsymbol{b}_i$. Now, it is easy to see that $\boldsymbol{h}_i^T\boldsymbol{x} \geq g_i \iff \boldsymbol{y}_i \in \mathbb{L}_+^2$, and

$$\|A_i\boldsymbol{x} + \boldsymbol{b}_i\| \leq \boldsymbol{c}_i^T\boldsymbol{x} + d_i \iff \boldsymbol{z}_i \in \mathbb{L}_+^{n_i+1}.$$

The inequalities and second order cone inequalities of Problem (1) can hence be reformulated as a set of equalities, and the generalized conic inequality

$$(\boldsymbol{y}_1, \dots, \boldsymbol{y}_p, \boldsymbol{z}_1, \dots, \boldsymbol{z}_m) \succeq_{K'} \boldsymbol{0}, \quad \text{where} \quad K' := (\mathbb{L}_+^2)^p \times \mathbb{L}_+^{n_1+1} \times \cdots \times \mathbb{L}_+^{n_m+1}$$

For the reduction of SDPs to the standard form ($\mathrm{P}_{\mathrm{Cone}}$), we need will need the following lemma:

Lemma 1. *The block diagonal matrix* $M = \begin{pmatrix} A_1 & & \\ & \ddots & \\ & & A_n \end{pmatrix}$ *is positive semidefinite if and only if each diagonal block is positive semidefinite:*

$$M \succeq 0 \iff \forall i \in [n],\ A_i \succeq 0.$$

Proof. The direct implication follows from the fact that the principal submatrices of a positive semidefinite matrix are positive semidefinite.

For the reverse implication, observe that

$$[\boldsymbol{u}_1^T, \dots, \boldsymbol{u}_n^T]M \begin{bmatrix} \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_n \end{bmatrix} = \sum_{i=1}^n \boldsymbol{u}_i^T A_i \boldsymbol{u}_i.$$

When the blocks A_i are positive semidefinite, the above sum has only nonnegative terms, so it is ≥ 0. Since the vectors $\boldsymbol{u}_1, \dots, \boldsymbol{u}_n$ were arbitrary, this proves $M \succeq 0$. □

Example:
The following problem is an SDP:

$$\begin{aligned}
\underset{\boldsymbol{x}\in\mathbb{R}^n,\ X\in\mathbb{S}^n}{\textbf{minimize}} \quad & \langle C, X\rangle + \boldsymbol{c}_0^T\boldsymbol{x} && \text{(2a)}\\
s.t. \quad & \langle F_i, X\rangle + \boldsymbol{f}_i^T\boldsymbol{x} = g_i, \quad (\forall i\in[p]) && \text{(2b)}\\
& \langle H_i, X\rangle + \boldsymbol{h}_i^T\boldsymbol{x} \geq \ell_i, \quad (\forall i\in[q]) && \text{(2c)}\\
& BXB^T + DX + X^TD \succeq R_1 && \text{(2d)}\\
& \sum_{i=1}^m x_iQ_i \succeq R_2. && \text{(2e)}
\end{aligned}$$

#3

The fact that the variables $\boldsymbol{x}$ and X are unconstrained can be handled as in Example #1. We assume that $R_1 \in \mathbb{S}^{r_1}$ and $R_2 \in \mathbb{S}^{r_2}$. We define the slack matrix

$$Z = \begin{pmatrix} \mathrm{Diag}(\boldsymbol{u}) & O & O \\ O & BXB^T + DX + X^TD - R_1 & O \\ O & O & \sum_{i=1}^m x_iQ_i - R_2 \end{pmatrix} \in \mathbb{S}^{q+r_1+r_2},$$

where $u_i = \langle H_i, X\rangle + \boldsymbol{h}_i^T\boldsymbol{x} - \ell_i$ $(\forall i\in[q])$. Now, by Lemma 1, the constraints (2c)-(2d)-(2e) are equivalent to $Z \succeq 0$, and we have only used linear equalities to introduce the slack variable Z.

2 Conic representability

In the next sections, we will see that many convex constraints of the form "$f(\boldsymbol{x}) \leq t$" can be reformulated as equivalent conic constraints. The next definition introduces the concept of K-representability, which should be interpreted as follows "an optimization problem involving the function f can be refomulated as a conic program over the cone K".

Definition 4. (Conic representability). Let $K \subseteq \mathbb{R}^d$ be a proper cone. A convex function $f : \mathbb{R}^n \to \mathbb{R}$ is *K-representable* if there exists a matrix $A \in \mathbb{R}^{p\times(n+1+m)}$ and a vector $\boldsymbol{b} \in \mathbb{R}^p$ such that

$$f(\boldsymbol{x}) \leq t \iff \exists \boldsymbol{u}\in\mathbb{R}^m :\ A\begin{bmatrix}\boldsymbol{x}\\ t\\ \boldsymbol{u}\end{bmatrix} \preceq_{\tilde{K}} \boldsymbol{b},$$

where $\tilde{K}$ is the direct product of several copies of K (i.e., $\tilde{K} = K^{\otimes k}$, where $k \in \mathbb{N}$ is such that $p = kd$). In other words, the constraint "$f(\boldsymbol{x}) \leq t$" is equivalent to a collection of k conic constraints "$F_j(\boldsymbol{x}, t, \boldsymbol{u}) \succeq_K \boldsymbol{0}$" for some affine functions F_j $(\forall j\in[k])$, that *may involve a set of $m \geq 0$ additional variables.*

Note: A concave function is said to be K-representable if $-f$ is K-representable.

We will see in a next lecture that conic programs can be solved efficiently for a number of cones, including $\mathbb{R}^n_+$, $\mathbb{L}^n_+$ and $\mathbb{S}^n_+$. Therefore, it is very important to understand which functions are K-representable, as it gives an answer to the following question: *given an efficient algorithm to solve conic programs over K, which class of optimization problems can be solved efficiently using this algorithm?*

Remark. Note however that in order to be used efficiently, we must restrict ourselves to functions with a *compact* conic representation, i.e., the number m of additional variables and the dimension p of the cone $\tilde{K}$ must be bounded by some polynomial in n.

We next review a few general *calculus rules* that preserve the notion of $K-$reprentability:

(a) Adding an affine part

If f is K-representable, then $g : \boldsymbol{x} \mapsto f(\boldsymbol{x}) + \boldsymbol{a}^T\boldsymbol{x} + b$ is K-representable.

(b) Composition with affine mapping

If f is K-representable, then $g : \boldsymbol{x} \mapsto f(A\boldsymbol{x} + \boldsymbol{b})$ is K-representable.

(c) Conic combinations

If $f_1, \ldots, f_k$ are K-representable and $\boldsymbol{\lambda} \geq \mathbf{0}$, then $g : \boldsymbol{x} \mapsto \sum_{i=1}^k \lambda_i f_i(\boldsymbol{x})$ is K-representable.

(d) Pointwise maximum

If $f_1, \ldots, f_k$ are K-representable then $g : \boldsymbol{x} \mapsto \max_{i=1}^k f_i(\boldsymbol{x})$ is K-representable.

(e) Inf-convolution

If $f_1, \ldots, f_k$ are K-representable then $g : \boldsymbol{x} \mapsto \inf\limits_{\substack{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_k \\ \boldsymbol{x}_1+\ldots+\boldsymbol{x}_k=\boldsymbol{x}}} f_1(\boldsymbol{x}_1)+f_2(\boldsymbol{x}_2)+\ldots+f_k(\boldsymbol{x}_k)$ is K-representable.

(f) Perspective transform

If f is K-representable, then $g(\boldsymbol{x}, t) = t\, f(\boldsymbol{x}/t)$, with $\mathbf{dom}\, g = \mathbf{dom}\, f \times \mathbb{R}_{++}$ is K-representable.

Proof. For (a), note that $g(\boldsymbol{x}) \leq t \iff f(\boldsymbol{x}) \leq t - \boldsymbol{a}^T\boldsymbol{x} - b$. Since f is K-representable, there exists an affine function F such that $f(\boldsymbol{x}) \leq t \iff F(\boldsymbol{x}, t, \boldsymbol{u}) \succeq_{\tilde{K}} \mathbf{0}$. But then,

$$g(\boldsymbol{x}) \leq t \iff G(\boldsymbol{x}, t, \boldsymbol{u}) := F(\boldsymbol{x}, t - \boldsymbol{a}^T\boldsymbol{x} - b, \boldsymbol{u}) \succeq_{\tilde{K}} \mathbf{0},$$

and G is affine (composition of two affine functions). (b) can be handled similarly:

$$g(\boldsymbol{x}) \leq t \iff H(\boldsymbol{x}, t, \boldsymbol{u}) := F(A\boldsymbol{x} + \boldsymbol{b}, t, \boldsymbol{u}) \succeq_{\tilde{K}} \mathbf{0},$$

where the function H is a composition of affine functions, hence affine. For (c), we assume that

$$f_i(\boldsymbol{x}) \leq t \iff \exists \boldsymbol{u}_i \in \mathbb{R}^m : \; A_i \begin{bmatrix} \boldsymbol{x} \\ t \\ \boldsymbol{u}_i \end{bmatrix} \preceq_{\tilde{K}_i} \boldsymbol{b}_i,$$

where $\tilde{K}_i$ is a direct product of copies of K. Then, it is easy to see that

$$\sum_{i=1}^k \lambda_i f_i(\boldsymbol{x}) \leq t \iff \exists \boldsymbol{u}_1, \ldots, \boldsymbol{u}_k \in \mathbb{R}^m, \exists t_1, \ldots, t_k \in \mathbb{R} : A_i \begin{bmatrix} \boldsymbol{x} \\ t_i \\ \boldsymbol{u}_i \end{bmatrix} \preceq_{\tilde{K}_i} \boldsymbol{b}_i \; (\forall i \in [k]), \text{ and } t_1 + \ldots + t_k \leq t.$$

Given any vector $\boldsymbol{u} \in K \setminus \mathbf{0}$, the inequality constraint $t_1 + \ldots + t_k \leq t$ can also be written as a conic inequality: $(t_1 + \ldots + t_k)\boldsymbol{u} \preceq_K t\boldsymbol{u}$, so it is possible to rewrite the condition $\sum_{i=1}^k \lambda_i f_i(\boldsymbol{x}) \leq t$ as a single big conic inequality (for a direct product of copies of K). The proof for (d) and (e) is similar. Finally, for the case of the perspective transformation (f), let f have a conic representation of the same form as in the definition. Then, for all $t > 0$,

$$g(x, t) \leq s \iff \exists \boldsymbol{u} : A \begin{bmatrix} \boldsymbol{x}/t \\ s/t \\ \boldsymbol{u} \end{bmatrix} \preceq_{\tilde{K}} \boldsymbol{b} \iff \exists \boldsymbol{u}' : A \begin{bmatrix} \boldsymbol{x} \\ s \\ \boldsymbol{u}' \end{bmatrix} \preceq_{\tilde{K}} t\boldsymbol{b} \iff \exists \boldsymbol{u}' : [A, -\boldsymbol{b}] \begin{bmatrix} \boldsymbol{x} \\ s \\ \boldsymbol{u}' \\ t \end{bmatrix} \preceq_{\tilde{K}} \mathbf{0}.$$

□

3 What can be expressed by Linear Programming ?

In this section, we will review a few nonlinear functions that are $\mathbb{R}^n_+$ representable, i.e., functions f such that the constraint

$$f(\boldsymbol{x}) \leq t$$

can be reformulated as a set of equivalent linear (in)equalities.

(a) Convex piecewise linear functions.

A convex piecewise linear function f can always be written as $f(\boldsymbol{x}) = \max_{i\in[m]} \boldsymbol{a}_i^T\boldsymbol{x} + b_i$ for some vectors $\boldsymbol{a}_i$'s and scalars b_i's. Then, the inequality $f(\boldsymbol{x}) \leq t$ is equivalent to

$$\boldsymbol{a}_i^T\boldsymbol{x} + b_i \leq t \quad (\forall i \in [m]).$$

(b) ℓ_1-norm of a vector:

$$\|\boldsymbol{x}\|_1 \leq t \iff \exists \boldsymbol{u} \in \mathbb{R}^n : \begin{cases} -\boldsymbol{u} \leq \boldsymbol{x} \leq \boldsymbol{u}; \\ \mathbf{1}^T\boldsymbol{u} \leq t. \end{cases}$$

(c) $\ell_\infty-$norm of a vector:

$$\|\boldsymbol{x}\|_\infty \leq t \iff (-t \leq x_i \leq t, \quad \forall i \in [n]).$$

(d) Sum of the k largest elements of a vector.

Denote by $s_k(\boldsymbol{x})$ the sum of the k largest elements of $\boldsymbol{x}$. The function $\boldsymbol{x} \mapsto s_k(\boldsymbol{x})$ is convex. Indeed, it can be rewritten as a pointwise maximum of linear functions:

$$s_k(\boldsymbol{x}) = \max_{1\leq i_1<i_2<\ldots<i_k\leq n} x_{i_1} + x_{i_2} + \ldots + x_{i_k}.$$

We could hence use the point (a) above to rewrite $s_k(\boldsymbol{x}) \leq t$ as a set of linear inequalities. However, this representation involves the maximum of $\binom{n}{k}$ functions. Below, we give a much more compact representation, thanks to the use of $n+1$ additional variables. Geometrically, we are in fact expressing the polytope $\{(\boldsymbol{x}, t) : s_k(\boldsymbol{x}) \leq t\} \subseteq \mathbb{R}^{n+1}$ as the projection of another polytope in $\mathbb{R}^{2(n+1)}$ over the $(\boldsymbol{x}, t)$-space:

$$s_k(\boldsymbol{x}) \leq t \iff \exists u \in \mathbb{R}, \exists \boldsymbol{v} \in \mathbb{R}^n : \begin{cases} ku + \mathbf{1}^T\boldsymbol{v} \leq t; \\ \boldsymbol{v} \geq \mathbf{0} \\ \boldsymbol{v} \geq \boldsymbol{x} - u\mathbf{1}. \end{cases}$$

Proof. We only prove the point (d), the other points are rather trivial. Denote by $x_{(1)} \geq x_{(2)} \geq \ldots \geq x_{(n)}$ the sorted elements of $\boldsymbol{x}$.

We first show that

$$s_k(\boldsymbol{x}) = \min_{u\in\mathbb{R}} \; ku + \sum_{i=1}^n \max(0, x_i - u). \tag{3}$$

Let u be any real number in the interval $[x_{(k+1)}, x_{(k)}]$. Then,

$$ku + \sum_{i=1}^n \max(0, x_i - u) = ku + \sum_{j=1}^k \big(x_{(j)} - u\big) = s_k(\boldsymbol{x}),$$

where the first equality comes from the fact that $j > k \implies x_{(j)} - u < 0$. This already shows that the optimal value $p^*(\boldsymbol{x})$ of the minimization problem in the right hand side of (3) satisfies $p^*(\boldsymbol{x}) \leq s_k(\boldsymbol{x})$.

On the other hand, for all $u \in \mathbb{R}$,

$$s_k(\boldsymbol{x}) = \sum_{j=1}^k x_{(j)} = \sum_{j=1}^k (x_{(j)} - u + u) = ku + \sum_{j=1}^k (x_{(j)} - u) \leq ku + \sum_{j=1}^k \max(0, x_{(j)} - u) \leq ku + \sum_{i=1}^n \max(0, x_i - u).$$

Therefore, $s_k(\boldsymbol{x}) \leq p^*(\boldsymbol{x})$. Finally, the representation of point (d) is obtained by introducing an auxiliary variable $v_i \geq \max(0, x_i - u)$ for all $i \in [n]$. □

Example:
We show how to reformulate the nonlinear optimization problem

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\textbf{minimize}} \quad & 3\,\|\boldsymbol{x}\|_1 + 7\,s_4(\boldsymbol{x}) \\ s.t. \quad & A\boldsymbol{x} \leq \boldsymbol{b} \end{aligned}$$

to an equivalent LP with $3n+1$ variables. We first use the epigraph representation

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n, \lambda_1, \lambda_2 \in \mathbb{R}}{\textbf{minimize}} \quad & 3\lambda_1 + 7\lambda_2 \\ s.t. \quad & A\boldsymbol{x} \leq \boldsymbol{b} \\ & \|\boldsymbol{x}\|_1 \leq \lambda_1 \\ & s_4(\boldsymbol{x}) \leq \lambda_2. \end{aligned}$$

Now, we use the points (b) and (d) above, which yields

$$\begin{aligned} \underset{\substack{\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}\in\mathbb{R}^n \\ t,\lambda_1,\lambda_2\in\mathbb{R}}}{\textbf{minimize}} \quad & 3\lambda_1 + 7\lambda_2 && \text{(4a)} \\ s.t. \quad & A\boldsymbol{x} \leq \boldsymbol{b} && \text{(4b)} \\ & -\boldsymbol{u} \leq \boldsymbol{x} \leq \boldsymbol{u} && \text{(4c)} \\ & \mathbf{1}^T\boldsymbol{u} \leq \lambda_1 && \text{(4d)} \\ & \boldsymbol{v} \geq \mathbf{0} && \text{(4e)} \\ & \boldsymbol{v} \geq \boldsymbol{x} - t\mathbf{1} && \text{(4f)} \\ & 4t + \mathbf{1}^T\boldsymbol{v} \leq \lambda_2 && \text{(4g)} \end{aligned}$$

#4

Finally, it is possible to eliminate λ_1 and λ_2 from this formulation, by putting the LHS of constraints (4d) and (4g) directly in the objective function:

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{u},\boldsymbol{v}\in\mathbb{R}^n,\ t\in\mathbb{R}}{\textbf{minimize}} \quad & 3\cdot(\mathbf{1}^T\boldsymbol{u}) + 7\cdot(4t + \mathbf{1}^T\boldsymbol{v}) \\ s.t. \quad & A\boldsymbol{x} \leq \boldsymbol{b} \\ & -\boldsymbol{u} \leq \boldsymbol{x} \leq \boldsymbol{u} \\ & \boldsymbol{v} \geq \mathbf{0} \\ & \boldsymbol{v} \geq \boldsymbol{x} - t\mathbf{1}. \end{aligned}$$

4 What can be expressed by Second Order Cone Programming ?

As in the previous section, $\boldsymbol{x}$ always denote a vector of dimension n. We review some nonlinear constraints involving the variable $\boldsymbol{x} \in \mathbb{R}^n$, as well as some scalar variables $t, y, z \in \mathbb{R}$, which can be handled by SOCP.

(a) Squared norm less than product of nonnegative variables

Inequalities of the form $\begin{cases} \|\boldsymbol{x}\|^2 \leq y \cdot z; \\ y \geq 0; \\ z \geq 0. \end{cases}$

The above system is equivalent to the SOC inequality

$$\left\| \begin{bmatrix} 2\boldsymbol{x} \\ y - z \end{bmatrix} \right\| \leq y + z.$$

Proof. This follows from

$$\left\| \begin{bmatrix} 2\boldsymbol{x} \\ y - z \end{bmatrix} \right\| \leq y + z \iff \begin{cases} \left\| \begin{bmatrix} 2\boldsymbol{x} \\ y - z \end{bmatrix} \right\|^2 & \leq & (y+z)^2; \\ y + z & \geq & 0. \end{cases}$$

$$\iff \begin{cases} 4\|\boldsymbol{x}\|^2 & \leq & (y+z)^2 - (y-z)^2; \\ y + z & \geq & 0. \end{cases} \qquad \iff \begin{cases} 4\|\boldsymbol{x}\|^2 & \leq & 4\, y \cdot z; \\ y, z & \geq & 0. \end{cases}$$

□

(b) Convex quadratic inequalities $\boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{a}^T \boldsymbol{x} + b \leq t$

By using the last point, we can reformulate *any* convex quadratic constraint as a SOC constraint. Indeed, the function $f : \boldsymbol{x} \mapsto \boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{a}^T \boldsymbol{x} + b$ is convex iff $Q \succeq 0$. It follows that we can compute a decomposition of the form $Q = H^T H$ (for example, by using a Cholesky decomposition). Then, we have $f(\boldsymbol{x}) = \|H\boldsymbol{x}\|^2 + \boldsymbol{a}^T \boldsymbol{x} + b$, and

$$f(\boldsymbol{x}) \leq t \iff \|H\boldsymbol{x}\|^2 \leq t - \boldsymbol{a}^T \boldsymbol{x} - b \iff \left\| \begin{bmatrix} 2H\boldsymbol{x} \\ t - \boldsymbol{a}^T \boldsymbol{x} - b - 1 \end{bmatrix} \right\| \leq t - \boldsymbol{a}^T \boldsymbol{x} - b + 1.$$

(c) Geometric and Harmonic means.

The geometric and harmonic means of a nonnegative vector are concave functions. We will show in the exercises that the inequalities

$$G(\boldsymbol{x}) = \prod_{i=1}^{n} x_i^{1/n} \geq t$$

and

$$H(\boldsymbol{x}) = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}} \geq t$$

can be represented by an equivalent system of SOC inequalities. (The harmonic mean is defined by continuity over all $\mathbb{R}^n_+$, with $H(\boldsymbol{x}) = 0$ whenever some $x_i = 0$.)

(d) Rational powers

The inequality $y^p \leq t$, where $p \in \mathbb{Q}, p \geq 1$ can be represented by a system of SOC inequalities. Indeed, let $p = \frac{\alpha}{\beta}$, with $\alpha \geq \beta \in \mathbb{N}$. Then,

$$x^{\frac{\alpha}{\beta}} \leq t \iff x^\alpha \leq t^\beta \iff x \leq G([\underbrace{t, \ldots, t}_{\beta \text{ times}}, \underbrace{1, \ldots, 1}_{(\alpha-\beta) \text{ times}}]).$$

Note: The point (b) shows that every convex QCQP (*quadratically constrained quadratic program*), that is, an optimization problem in which the objective function and the constraints are convex quadratic functions, can be cast as a SOCP.

5 What can be expressed by Semidefinite Programming ?

Before we start to enumerate nonlinear functions that can be handled by SDP (i.e., $\mathbb{S}^n_+$-representable functions), we state a very important lemma, which is at the heart of most SDP representations. We will prove this result in the exercises.

Lemma 2 (Schur Complement). *Let M be a symmetric matrix paritioned in blocks as $M = \begin{bmatrix} A & C \\ C^T & B \end{bmatrix}$. The following holds:*

$$M \succ 0 \iff \begin{cases} B \succ 0 \\ A - CB^{-1}C^T \succ 0 \end{cases} \iff \begin{cases} A \succ 0 \\ B - C^T A^{-1} C \succ 0 \end{cases}$$

Moreover, if $B \succ 0$, then

$$M \succeq 0 \iff A - CB^{-1}C^T \succeq 0;$$

And similarly, if $A \succ 0$, then

$$M \succeq 0 \iff B - C^T A^{-1} C \succeq 0.$$

Note: When the matrix B is singular, there is an extended version of the lemma which characterizes the positive semidefiniteness of M in terms of the Schur complement $A - CB^\dagger C^T$, where $B^\dagger$ is the Moore-Penrose pseudo inverse of B; In this case, an additional technical condition is required:

$$\begin{aligned} M \succeq 0 &\iff B \succeq 0 \quad \text{and} \quad \mathrm{A} - \mathrm{C}\mathrm{B}^\dagger\mathrm{C}^\mathrm{T} \succeq 0 \quad \text{and} \quad \mathbf{Im}\,\mathrm{C}^\mathrm{T} \subseteq \mathbf{Im}\,\mathrm{B}. \\ &\iff A \succeq 0 \quad \text{and} \quad \mathrm{B} - \mathrm{C}^\mathrm{T}\mathrm{A}^\dagger\mathrm{C} \succeq 0 \quad \text{and} \quad \mathbf{Im}\,\mathrm{C} \subseteq \mathbf{Im}\,\mathrm{A}. \end{aligned}$$

In what follows, $X \in \mathbb{S}^n$ is a matrix variable, $\boldsymbol{x} \in \mathbb{R}^m$ is a vector of variable, and $t \in \mathbb{R}$ is a scalar variable. We review some nonlinear functions such that $f(X) \leq t$ or $g(\boldsymbol{x}) \leq t$ can be represented by a LMI (linear matrix inequality).

(a) Second Order Cone Programming

The SOC inequality $\|\boldsymbol{x}\| \leq t$ can be cast as a LMI. Indeed,

$$\|\boldsymbol{x}\| \leq t \iff \begin{cases} \|\boldsymbol{x}\|^2 \leq t^2 \\ t \geq 0 \end{cases} \iff \begin{cases} \boldsymbol{x}^T (tI_n)^{-1} \boldsymbol{x} \leq t \\ t \geq 0 \end{cases} \iff \begin{bmatrix} t\,I_n & \boldsymbol{x} \\ \boldsymbol{x}^T & t \end{bmatrix} \succeq 0.$$

This shows that any SOCP can be cast as a SDP. This is often not a good idea in practice, but this shows that SDP is a *superclass* of SOCP !

(b) Let $\boldsymbol{c} \in \mathbb{R}^n$, and let $K : \mathbb{R}^m_+ \to \mathbb{S}^n_{++}$ be an affine function. Then, the Schur complement lemma tells us that

$$\boldsymbol{c}^T K(\boldsymbol{x})^{-1} \boldsymbol{c} \leq t \iff \begin{bmatrix} K(\boldsymbol{x}) & \boldsymbol{c} \\ \boldsymbol{c}^T & t \end{bmatrix} \succeq 0.$$

(c) Largest eigenvalue of a matrix.

Let $M : \mathbb{R}^m \to \mathbb{S}^n$ be an affine function, that is, $M(\boldsymbol{x}) = M_0 + \sum_{i=1}^m x_i M_i$ for some $M_i \in \mathbb{S}^n$. Then,

$$\lambda_{\max}(M(\boldsymbol{x})) \leq t \iff M(\boldsymbol{x}) \preceq t\,I_n.$$

By the way, this shows that $\lambda_{\max}$ is a convex function over $\mathbb{S}^n$.

(d) Smallest eigenvalue of a matrix.

Similarly, $\lambda_{\min}$ is a concave function over $\mathbb{S}^n$, and

$$\lambda_{\min}(M(\boldsymbol{x})) \geq t \iff M(\boldsymbol{x}) \succeq t\,I_n.$$

(e) nth root of determinant.

The nth root of the determinant is a concave function over $\mathbb{S}^n_{++}$. We have the following representation with LMIs:

$$(\det X)^{\frac{1}{n}} \geq t \iff \exists L \in \mathbb{R}^{n\times n}, \boldsymbol{u} \in \mathbb{R}^n : \begin{cases} \begin{bmatrix} X & L \\ L^T & \operatorname{Diag} \boldsymbol{u} \end{bmatrix} \succeq 0; \\ L \text{ is Lower triangular;} \\ \mathbf{diag}\, L = \boldsymbol{u}; \\ G(\boldsymbol{u}) \geq t, \end{cases}$$

where $G(\boldsymbol{u})$ is the geometric mean of the vector $\boldsymbol{u}$ (cf. Section 4, Point (c)).

Proof. (a) and (b) are already proved, and (d) is similar to (c). We only need to prove (c) and (e).

(c) The largest eigenvalue of a matrix satisfies the variational characterization

$$\lambda_{\max}(X) = \sup_{\boldsymbol{u}\in\mathbb{R}^n\setminus\{\mathbf{0}\}} \frac{\boldsymbol{u}^T X \boldsymbol{u}}{\boldsymbol{u}^T \boldsymbol{u}}.$$

Hence,

$$\begin{aligned} \lambda_{\max}(X) \leq t &\iff \forall \boldsymbol{u} \in \mathbb{R}^n \setminus \{\mathbf{0}\}, \quad \frac{\boldsymbol{u}^T X \boldsymbol{u}}{\boldsymbol{u}^T \boldsymbol{u}} \leq t \\ &\iff \forall \boldsymbol{u} \in \mathbb{R}^n, \quad \boldsymbol{u}^T X \boldsymbol{u} \leq t \boldsymbol{u}^T \boldsymbol{u} \\ &\iff \forall \boldsymbol{u} \in \mathbb{R}^n, \quad \boldsymbol{u}^T (X - tI_n) \boldsymbol{u} \leq 0 \\ &\iff (X - tI_n) \preceq 0. \end{aligned}$$

(e) $\implies$. Let $X \succ 0$, and let $X = JJ^T$ be a Cholesky decomposition of X. Let D be the diagonal matrix with diagonal elements u_i, where $u_i := J_{ii}^2$ is the square of the ith diagonal element of J, and let $L = JD^{1/2}$, so $L_{ii} = J_{ii}^2 = D_{ii} = u_i$. We have $X = LD^{-1}L^T$, so the Schur-complement lemma tells us that

$$\begin{bmatrix} X & L \\ L^T & D \end{bmatrix} \succeq 0.$$

By construction, L is lower triangular, and the diagonal elements of L are the same as the diagonal elements u_i of D. Finally, since the determinant of a triangular matrix is equal to the product of its diagonal elements, we have

$$\det X = \det L \det D^{-1} \det L = \left(\prod u_i\right)\left(\prod u_i\right)^{-1}\left(\prod u_i\right) = \left(\prod u_i\right).$$

Hence, $(\det X)^{\frac{1}{n}} \geq t \implies G(\boldsymbol{u}) \geq t$.

$\Longleftarrow$. We first prove the intermediate result $A \succeq B \succeq 0 \implies \det A \geq \det B$. First note that the statement is trivial whenever B is singular. So we assume $B \succ 0$. If $M \succeq I$, then we know from point (d) that the eigenvalues of M are larger than 1, so $\det M \geq 1$. Applying this to the matrix $M = B^{-1/2}AB^{-1/2}$ yields $A \succeq B \implies B^{-1/2}AB^{-1/2} \succeq I \implies \det B^{-1} \det A \geq 1$, hence the desired result.

Now, if the system of constraints holds and $t > 0$, then by the Schur complement lemma, we have $X \succeq L \operatorname{Diag} \boldsymbol{u}^{-1} L^T$. So, using the implication $A \succeq B \implies \det A \geq \det B$, we obtain $\det X \geq (\prod u_i)(\prod u_i)^{-1}(\prod u_i) = (\prod u_i)$, and $(\det X)^{\frac{1}{n}} \geq G(\boldsymbol{u}) \geq t$. In the trivial case $t \leq 0$, the LMI tells us that $X \succeq 0$, hence $\det X \geq 0 \geq t$. □

6 The Exponential Cone

Clearly, the cones seen so far only allow to handle *semi-algebraic functions*, i.e., functions such that the equation $f(\boldsymbol{x}) \leq t$ is equivalent to a set of polynomial inequalities in $\boldsymbol{x}$, t (and eventually additional variables $\boldsymbol{u}$). Indeed, linear and second-order cone inequalities are special cases of LMI, and by using Silverster criterion (a matrix is positive semidefinite iff all its principal minors are nonnegative), we can rewrite any LMI as a (large) set of polynomial inequalities.

This rules out the possibility of existence of an LMI for simple convex constraints involving transcendental functions, such as $e^x \leq t$. But we may still ask ourselves whether such constraints can be represented as conic inequalities. Obviously, only convex functions can be conic representable:

Proposition 3. *If f is K-representable for some proper cone K, then f is convex.*

Proof. If f is K-representable, then its epigraph is (the projection over a subset of coordinates of) the reverse affine image of a proper (hence convex) cone. So, $\mathbf{epi}\, f$ is convex, hence f must be convex. □

A general result gives a (partial) converse to this statement: every convex function is K-representable for the cone K obtained by taking the perspective transformation of its epigraph (with technical conditions to ensure the cone is proper). We leave the proof of this statement as an exercise.

Theorem 4. *Let f be a convex function, with $\mathbf{int}\,\mathbf{dom}\, f \neq \emptyset$ and such that $\mathbf{epi}\, f$ does not contain any line (which basically means that there is no direction along which f is linear), and define*

$$\tilde{K} = \{(\boldsymbol{x}, y, z) \in \mathbb{R}^{n+2} : y > 0, \quad yf(x/y) \leq z\}.$$

Then, $K = \mathbf{cl}\,\tilde{K}$ is a proper cone, and f is K-representable:

$$f(\boldsymbol{x}) \leq t \iff (\boldsymbol{x}, 1, t) \succeq_K \mathbf{0}.$$

A cone of particular interest is the *exponential cone*, which is obtained as in the above theorem, for the exponential function:

$$\begin{aligned} K_{\exp} &:= \mathbf{cl}\,\{(x,y,z) \in \mathbb{R}^3 :\ y > 0,\ ye^{x/y} \leq z\} \\ &= \{(x,y,z) \in \mathbb{R}^3 :\ y > 0,\ ye^{x/y} \leq z\} \cup \{(x,y,z) | x \leq 0, y = 0, z \geq 0\}. \end{aligned}$$

We will see in the exercises that this cone allows to give a conic representation for two important convex functions, namely:

(a) The relative entropy (or Kullback-Leibler divergence) of two nonnegative vectors $\boldsymbol{x}, \boldsymbol{y} \geq \mathbf{0}$:

$$f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1^n} x_i \log \frac{x_i}{y_i},$$

with the appropriate convention to "close the function f" by continuity at the boundary of $\mathbb{R}^{2n}_+$: For all $\alpha > 0$, $0 \log 0/\alpha := 0$, $0 \log 0/0 := 0$ and $\alpha \log \alpha/0 := \infty$.

(b) The log-sum-exp function, $f(\boldsymbol{x}) = \log(\sum_{i=1}^n e^{x_i})$.

An important application of optimization with log-sum-exp functions is *Geometric Programming* (GP). A function of the form $f : \mathbb{R}^n_{>0} \mapsto \mathbb{R}$, $f(\boldsymbol{x}) = \alpha \boldsymbol{x}_1^{p_1} \cdots \boldsymbol{x}_n^{p_n}$, where $\alpha > 0$ and the p_i's are real-valued exponents is called a (generalized) *monomial*, and a sum of monomials is a *posynomial*. For example, $f(x,y,z) = \frac{\sqrt{x}}{y^{2.5}z} + 2zy^{1.7} + 6$ is a posynomial, but $g(x,y,z) = 2x^{-1.3}z^{1.4} - 3x^{2.1}y^{1.6}$ is not (because of the coef -3). An optimization problem of the form

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathbb{R}^n_{++}} \quad & f_0(\boldsymbol{x}) \\ s.t. \quad & f_i(\boldsymbol{x}) \leq 1,\ (i = 1, \ldots, m) \\ & h_j(\boldsymbol{x}) = 1,\ (j = 1, \ldots, p), \end{aligned}$$

where the f_i's are posynomials and the h_i's are monomials, is called a *geometric program* (in posynomial form). Geometric programming has many applications, in particular in electrical engeneering. In fact, we shall see that every GP can be reformulated as an equivalent exponential cone program, i.e., a conic program over $K_{\exp}$. Taking the logarithm of the objective function and constraints, the GP is equivalent to:

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathbb{R}^n_{++}} \quad & \log f_0(\boldsymbol{x}) \\ s.t. \quad & \log f_i(\boldsymbol{x}) \leq 0,\ (i = 1, \ldots, m) \\ & \log h_j(\boldsymbol{x}) = 0,\ (j = 1, \ldots, p). \end{aligned}$$

Now, we perform the change of variable $x_i = e^{y_i}$. Then, the function $\log f_i(x)$ become a log-sum-exp with affine substitution of the argument. To see this, consider a posynomial $f(\boldsymbol{x}) = \sum_k \alpha_k \prod_i x_i^{p_{ik}}$. We have:

$$\log f(\boldsymbol{x}) = \log\left(\sum_k \alpha_k \prod_i e^{p_{ik}\, y_i}\right) = \log\big(\sum_k \exp(\log\alpha_k + \sum_i y_i p_{ik})\big).$$

For the case of an monomial $h(\boldsymbol{x}) = \beta \prod_i x_i^{q_i}$, the log-transformation yields a linear function:

$$\log h(\boldsymbol{x}) = \log\beta + \sum_i q_i y_i.$$

As for $\mathbb{R}^n_+$, $\mathbb{L}^n_+$ and $\mathbb{S}^n_+$, we will see in a next lecture that conic programs over $K_{\exp}$ can be solved efficiently. However, we point that, unlike the three other cones, $K_{\exp}$ is not self-dual. This has a number of drawbacks for algorithmic purposes.

Theorem 5. *The cones $\mathbb{R}^n_+$, $\mathbb{L}^n_+$ and $\mathbb{S}^n_+$ are self-dual, i.e., $(\mathbb{R}^n_+)^* = \mathbb{R}^n_+$, $(\mathbb{L}^n_+)^* = \mathbb{L}^n_+$ and $(\mathbb{S}^n_+)^* = \mathbb{S}^n_+$. This is not the case for the exponential cone, whose dual is the relative entropy cone:*

$$K^*_{\exp} = \{(u,v,w) \in \mathbb{R}^3 :\ u \le 0,\ w \ge 0,\ -u\log(-u/w) \le v - u\},$$

where $0\log 0 := 0$.

Proof. 1) $K = \mathbb{R}^n_+$. Let $\boldsymbol{x} \in \mathbb{R}^n_+$. Then, for all $\boldsymbol{y} \in \mathbb{R}^n_+$, $\boldsymbol{x}^T\boldsymbol{y}$ is a sum of nonnegative terms, hence $\boldsymbol{x}^T\boldsymbol{y} \ge 0$. This shows $\mathbb{R}^n_+ \subseteq (\mathbb{R}^n_+)^*$. For the reverse inclusion, assume $\boldsymbol{x} \not\ge \boldsymbol{0}$, i.e., there is a coordinate $i \in [n]$ such that $x_i < 0 \iff \langle \boldsymbol{x}, \boldsymbol{e}_i\rangle < 0$. Since $\boldsymbol{e}_i \in \mathbb{R}^n_+$, this shows $\boldsymbol{x} \notin (\mathbb{R}^n_+)^*$.

2) $K = \mathbb{L}^n_+$. Let $\|\boldsymbol{x}\| \le t$. Then, for all $\boldsymbol{y}$ such that $\|\boldsymbol{y}\| \le s$, we have $\langle(\boldsymbol{x},t),(\boldsymbol{y},s)\rangle = \boldsymbol{x}^T\boldsymbol{y} + ts \ge -\|\boldsymbol{x}\|\|\boldsymbol{y}\| + ts \ge ts - ts = 0$, where the first inequality is Cauchy Schwarz. This shows $\mathbb{L}^n_+ \subseteq (\mathbb{L}^n_+)^*$. For the converse inclusion, assume $(\boldsymbol{x},t) \notin \mathbb{L}^n_+$, i.e., $\|\boldsymbol{x}\| > t$. Then, define $\boldsymbol{y} = -\boldsymbol{x}$ and $s = \|\boldsymbol{y}\| = \|\boldsymbol{x}\| > t$, so $(\boldsymbol{y},s) \in \mathbb{L}^n_+$. We have $\langle(\boldsymbol{x},t),(\boldsymbol{y},s)\rangle = \boldsymbol{x}^T\boldsymbol{y} + ts = -\|\boldsymbol{y}\|^2 + ts = -s^2 + ts = s(t-s) < 0$. Hence, $(\boldsymbol{x},t) \notin (\mathbb{L}^n_+)^*$.

3) $K = \mathbb{L}^n_+$. Let $X \succeq 0$. Then, $X = LL^T$ for some matrix L, and for all $Y \succeq 0$, there exists a matrix H such that $Y = HH^T$. Hence, $\langle X, Y\rangle = \operatorname{trace} LL^THH^T = \operatorname{trace} H^TLL^TH = \|H^TL\|_F^2 \ge 0$. This shows $\mathbb{S}^n_+ \subseteq (\mathbb{S}^n_+)^*$. For the reverse inclusion, assume $X \not\succeq 0$, i.e., $\exists \boldsymbol{u} : \boldsymbol{u}^TX\boldsymbol{u} < 0 \iff \langle X, \boldsymbol{u}\boldsymbol{u}^T\rangle < 0$. Since $\boldsymbol{u}\boldsymbol{u}^T \in \mathbb{S}^n_+$, this shows $X \notin (\mathbb{S}^n_+)^*$.

4) $K = K_{\exp}$. Denote by K_r the cone defined in the theorem. To show $K_r \subseteq K^*_{\exp}$, we will demonstrate that the inclusion holds for vectors in the interior of these cones, i.e., when all inequalities hold strictly. So, let (x,y,z,u,v,w) satisfy $y > 0,\ ye^{\frac{x}{y}} < z, u < 0, w > 0, -u\log(\frac{-u}{w}) < v - u$. The last inequality can be rewritten as $w > -ue^{v/u-1}$. Hence, $\langle(x,y,z),(u,v,w)\rangle = xu + yv + zw \ge xu + yv - yue^{x/y+v/u-1} = -yu(-x/y - v/u + e^{x/y+v/u-1})$. This expression is the product of $y \ge 0$, $-u \ge 0$, and $e^A - (1+A) \ge 0$, where we have set $A = x/y + v/u - 1$. So, this shows $\mathbf{int}\, K_r \subseteq (\mathbf{int}\, K_{\exp})^*$, and the inclusion $K_r \subseteq K^*_{\exp}$ follows by taking the closure.

For the converse inclusion, let $(u,v,w) \notin K_r$. Here, we must distinguish several cases:

a) if $u > 0$, then we consider $(x,y,z) = (-1,0,0) \in K_{\exp}$, so $\langle(x,y,z),(u,v,w)\rangle = -u < 0$;

b) if $w < 0$, then we take $(x,y,z) = (0,0,1) \in K_{\exp}$, so $\langle(x,y,z),(u,v,w)\rangle = w < 0$;

c) The remaining case is $u \le 0, w \ge 0, -u\log(-u/v) > v - u$. Here, we must again distinguish the subcases $(u < 0, w > 0)$, $(u = 0)$, and $(u < 0, w = 0)$. For $u, w \ne 0$ we consider the vector $(x, 1, e^x) \in K_{\exp}$ for $x = \log(-u/w)$, so $\langle(x,y,z),(u,v,w)\rangle = u\log(-u/w) + v - u < 0$. For $u = 0$ with the convention $0\log 0 = 0$, the condition $-u\log(-u/v) > v - u$ rewrites $v < 0$. Then, $\langle(x,1,e^x),(u,v,w)\rangle = v + we^x$ is negative for $x \to -\infty$. And for $(u < 0, w = 0)$, we have $\langle(x,1,e^x),(u,v,w)\rangle = xu + v$, which is negative for $x > v/(-u)$.

In all cases, we were able to find a vector $(x,y,z) \in K_{\exp}$ such that $\langle(x,y,z),(u,v,w)\rangle < 0$, so $(u,v,w) \notin K^*_{\exp}$.

□

7 A very quick guide through the PICOS interface

The python interfaces PICOS and CVXPY allow the users to easily implement conic programming problems in python, and to solve them with state-of-the-art solvers.

We give a few details on the PICOS syntax below:

- To create a problem instance:

```
import picos
P = picos.Problem()
```

- Variables can be added to the problem as follows:

```
x = P.add_variable(name, size=(1,1), type="continuous")
```

  Here, `<name>` is a string, `<size>` is an integer (for vectors) or a pair of integers (for matrices), and `<type>` can be used to indicate that the variable has a special type, for example `type="symmetric"` indicates that the variable is a symmetric matrix. Other interesting types are `"binary"` and `"integer"` for (mixed)-integer optimization, but this goes out of the scope of this lecture.

- The objective function should be specified by

```
P.set_objective(direction, objective)
```

  where `<direction>` is either `"max"` or `"min"`, and `<objective>` is an *affine expression* formed with the variables of the problem.

- Constraints are added with the syntax

```
P.add_constraint(cons)
```

  where `<cons>` is a *constraint* obtained by comparing two *expressions*, for example

```
P.add_constraint(expression1 <= expression2)
P.add_constraint(expression1 == expression2)
```

  for inequality and equality constraints, respectively.

- When the problem is fully implemented, we can solve it with

```
P.solve()
```

  This calls the most appropriate solver, solves the problem, and stores the optimal value of the variables in their `value` attribute. Optimal dual variables of the constraints are computed, too (cf. next chapter), and stored in the `dual` attribute of the constraints.

- To enter constraints, the following operators are useful:
  - The standard python operators for the usual operations are `+`, `-`, `*`, `/` and `**` for exponentiation.
  - The operators `<<` and `>>` are used to denote the conic order relative $\mathbb{S}^n_+$.
  - The operator `|` stands for the scalar product. For example, `1|x` is understood as the sum of all elements of `x`.
  - The operator `abs()` stands for the Euclidean (or Frobenius) norm of its argument.
  - The operator `&` can be used to concatenate matrix blocks, horizontally.
  - The operator `//` can be used to concatenate matrix blocks, vertically.
  - The operator `picos.diag` applied to a vector $\boldsymbol{u}$ returns the matrix $\mathrm{Diag}(\boldsymbol{u})$.
  - The operator `picos.diag_vect` applied to a matrix X returns the vector $\mathbf{diag}\,(X)$.
  - The property `.T` transposes a vector/matrix.
  - The operator `picos.sum` applied to a *list of affine expressions*, returns the sum of all expressions in the list.

- PICOS natively supports most of the nonlinear functions listed in this chapter. For example, the constraint $x^{2.7} <= t$ can be entered *as is* in PICOS, which automatically translates `x**2.7 <= t` as a set of equivalent SOC-constraints (cf. Section 4, point (d)). Also, the constraints $\|\boldsymbol{x}\|^2 \leq yz,\ y,z \geq 0$ can be entered in picos directly as

  ```
  P.add_constraint(abs(x)**2 <= y*z)
  ```

  For this special case, PICOS understands that we mean the implicit constraints $y \geq 0, z \geq 0$, and automatically reformulates the constraint as $\left\| \begin{bmatrix} 2\boldsymbol{x} \\ y-z \end{bmatrix} \right\| \leq y+z$. In other words, the above code fragment is equivalent to:

  ```
  P.add_constraint(abs( ((2*x) // (y-z)) ) <= y*z )
  ```

- Here are a few examples of nonlinear functions implemented in PICOS:
  - `picos.sum_k_largest(x,k) <= t` represents the constraint $s_k(\boldsymbol{x}) \leq t$;
  - `picos.norm(x,p) <= t` represents the constraint $\|\boldsymbol{x}\|_p \leq t$;
  - `picos.geomean(x) >= t` represents the constraint $G(\boldsymbol{x}) = \prod x_i^{1/n} \geq t$;
  - `picos.detrootn(X) >= t` represents the constraint $(\det X)^{1/n} \geq t$;
  - `picos.lambda_max(X) <= t` represents the constraint $\lambda_{\max}(X) \leq t$.
- The exponential cone is now supported in PICOS (thanks to Maximilan Stahlberg ☺):
  - `picos.exp(x) <= t` represents the constraint $e^x \leq t$.
  - `picos.logsumexp(x) <= t` represents the constraint $\log(\sum_i e^{x_i}) \leq t$.
  - `picos.kullback_leibler(x,y) <= t` represents the constraint $\sum_i x_i \log \frac{x_i}{y_i} \leq t$.

Example:
To illustrate the above rules, let us implement the following (dummy) problem in PICOS

$$\begin{aligned} \min_{X\in\mathbb{S}^n, \boldsymbol{x}\in\mathbb{R}^m, t\in\mathbb{R}} \quad & \langle A, X\rangle + \boldsymbol{b}^T\boldsymbol{x} + 3t \\ s.t. \quad & \begin{bmatrix} X & \boldsymbol{x} \\ \boldsymbol{x}^T & t \end{bmatrix} \succeq 0 \\ & \mathrm{Diag}(X) = \boldsymbol{x} \\ & \mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1} \\ & \mathbf{1}^T\boldsymbol{x} = 1 \\ & X_{3,3} + X_{1,2} + X_{2,1} + 2\,x_6 \leq x_8 \\ & \|C\boldsymbol{x} - \boldsymbol{d}\| \leq t \end{aligned}$$

We assume that the data $A \in \mathbb{S}^n$, $\boldsymbol{b} \in \mathbb{R}^m$, $C \in \mathbb{R}^{r\times m}$ and $\boldsymbol{d} \in \mathbb{R}^r$ is already loaded in memory, and stored in objects `A,b,C,d` of the class `cvxopt.matrix` (the package `cvxopt` is a dependency of PICOS).

#5

```
#Define the problem and the variables
P = picos.Problem()
X = P.add_variable('X',(n,n),'symmetric')
x = P.add_variable('x',m)
t = P.add_variable('t',1)

#add the constraints
P.add_constraint( ( (X & x)//(x.T & t) ) >> 0)
P.add_constraint( picos.diag_vect(X) == x )
P.add_constraint( x >= 0)                  #(here, PICOS understands >= is elementwise)
P.add_constraint( x <= 1)
P.add_constraint( (1|x) == 1)              #(here, '1' is recognized as the vector of ones)
P.add_constraint( X[2,2] + X[0,1] + X[1,0] + 2*x[5] <= x[7] )     #(indices start from 0)
P.add_constraint( abs(C*x-d) <= t)

#set the objective function and solve the problem
P.set_objective('min',(A|X) + (b.T*x) + 3*t)
P.solve()
```