Identifying Redundant Linear Constraints in Systems of Linear Matrix Inequality Constraints

Shafiu Jibrin (`shafiu.jibrin@nau.edu`)
Department of Mathematics and Statistics
Northern Arizona University, Flagstaff
Arizona 86011-5717

Daniel Stover (`dbs23@dana.ucc.nau.edu`)
Department of Mathematics and Statistics
Northern Arizona University, Flagstaff
Arizona 86011-5717

Abstract. Semidefinite programming has been an interesting and active area of research for several years. In semidefinite programming one optimizes a convex (often linear) objective function subject to a system of linear matrix inequality constraints. Despite its numerous applications, algorithms for solving semidefinite programming problems are restricted to problems of moderate size because the computation time grows faster than linear as the size increases. There are also storage requirements. So, it is of interest to consider how to identify redundant constraints from a semidefinite programming problem. However, it is known that the problem of determining whether or not a linear matrix inequality constraint is redundant or not is NP complete, in general.

In this paper, we develop deterministic methods for identifying all redundant *linear* constraints in semidefinite programming. We use a characterization of the normal cone at a boundary point and semidefinite programming duality. Our methods extend certain redundancy techniques from linear programming to semidefinite programming.

Keywords: Semidefinite programming, Linear matrix inequalities, Redundancy, Normal cone.

AMS Subject Classification: 90C25, 15A45, 15A39, 15A48

1. Introduction

Semidefinite Programming is one of the most interesting and challenging areas in operations research to emerge in the last decade. Semidefinite programming problems arise in a variety of applications e.g., in engineering, statistics and combinatorial optimization ([3], [10]. A semidefinite programming problem is an extension of a linear programming problem where the linear constraints are replaced by linear matrix inequality (LMI) constraints. Semidefinite programming also generalizes quadratically constrained quadratic programming QCQP ([24]). We consider semidefinite programming program (SDP) of the following form:

$$\begin{aligned} SDP:\ min\ \ & c^T x & (1.1)\\ s.t.\qquad & A(x) \succeq 0 \\ & b_j + a_j^T x \geq 0\ (j = 1, 2, \dots, q) \end{aligned}$$

where the variable is $x \in \mathbb{R}^n$ and

$$A(x) := A_0 + \sum_{i=1}^{n} x_i A_i.$$

The problem data are the $m \times m$ symmetric matrices A_i and the vectors a_j, b_j, c in $\mathbb{R}^n$. The variable is the vector $x \in \mathbb{R}^n$. The constraint $A(x) \succeq 0$ is called a *linear matrix inequality* (LMI) and $A(x) \succeq 0$ means that all the eigenvalues of $A(x)$ are nonnegative.

Nestrov and Nemirovsky in 1988 [21] showed that interior-point methods for linear programs can, in principle, be generalized to convex optimization problems such as SDP's. Many polynomial time interior point methods have been developed for solving SDP's including ([3], [8], [25], [23]). A lot is known on both the classes of problems that SDP formulation can handle, and the best SDP algorithms to solve each class. In addition, a great deal has been learned about the limits of the current algorithms to solving SDP's.

One limitation to current SDP algorithms is that they are restricted to problems where the number of constraints are moderate [5]. This is due to the fact that the number of computations required becomes too high for large problems. In practice, the number of iterations required to solve SDP problems grows very slowly with the size of the problem [24].

The most important factor in the time complexity is the cost per iteration. For example, each iteration of the primal-dual algorithm described in [24] requires $\mathbf{O}(n^2(m+q)^2)$ time. Significant savings in computation time can be gained if the number of constraints can be effectively reduced before the problems are solved. Furthermore, storage requirements are at least as important as the computational time complexity [9]. The second most important issue in creating a fast implementation of an interior point method is preprocessing to reduce problem size [20].

A *redundant* constraint is one that can be removed without changing the feasible region defined by the original system of constraints, otherwise it is called *necessary*. Larger SDP problems can be solved if we can identify and eliminate the redundant constraints before the problems are solved. There is always a positive result from identifying redundancy [18]. Apart from computational and storage difficulties caused by redundancy, the knowledge that a constraint is redundant might offer additional insight into the problem model or might lead to different decisions about the problem. So, it is of interest to study how to identify redundant constraints. An interesting question that naturally arises is: Can we extend all redundancy methods from linear programming to semidefinite programming? The answer is no. It has been shown that the problem of determining whether or not an LMI constraint is redundant in semidefinite programming is NP complete [17], in general.

We are concerned with identifying all redundant linear constraints from the SDP (1.1). A naive approach would be to identify all the redundant linear constraints with respect to the system of the linear constraints only. Such approach would only guarantee a partial identification of all the redundant linear constraints in SDP 1.1. We assume that there is a large number of linear constraints in the problem, that is, q is large.

There is abundant literature on redundancy techniques in the case of linear programming. An excellent survey of these is given in ([11],[18]). One class of the methods is called *hit-and-run*. These methods are probabilistic and they work by detecting the necessary constraints. There is a small chance that a constraint declared redundant by a hit-and-run method, is actually necessary. In ([17], [16]), hit-and-run redundancy detection methods were extended from linear programming to semidefinite programming.

A different and relatively more expensive class of redundancy methods in linear programming are the deterministic methods. An early paper on these methods is that of

Balinsky [4], published in 1961. See also ([18], [11] and [12]). Deterministic methods have the advantage that they identify both necessary and redundant linear constraints with probability 1. The study of deterministic methods for identifying redundant constraints in semidefinite programming seems to be somewhat absent from the literature. A study of presolving for SDP is given in [14].

In this paper, we develop deterministic methods for the identification of all redundant linear constraints from (1.1). We use SDP duality and a characterization of the normal cone at a boundary point. Our methods generalize some of the redundancy techniques in linear programming to semidefinite programming. This work gives practical methods for identifying all redundant linear constraints in semidefinite programming.

2. Identifying Redundant Linear Constraints

In this section, we present our main result for identifying all redundant linear constraints from the system of inequalities that defines the feasible region of SDP (1.1). We consider the system:

$$A(x) \succeq 0 \tag{2.1}$$

$$b_j + a_j^T x \geq 0 \ (j = 1, 2, \ldots, q) \tag{2.2}$$

Let

$$\mathcal{R} = \{x \in \mathbb{R}^n : A(x) \succeq 0,\ b_j + a_j^T x \geq 0,\ j = 1, 2, \ldots, q\} \tag{2.3}$$

$$\mathcal{R}_k = \{x \in \mathbb{R}^n : A(x) \succeq 0,\ b_j + a_j^T x \geq 0,\ j = 1, \ldots, k-1, k+1, \ldots, q\} \tag{2.4}$$

The *kth* linear constraint is said to be *redundant* with respect to $\mathcal{R}$ if $\mathcal{R} = \mathcal{R}_k$. Otherwise, it is said to be *necessary.* It is called *weakly redundant* if it is redundant and its boundary $\{x \in \mathbb{R}^n : b_k + a_k^T x = 0\}$ touches the feasible region $\mathcal{R}$. A linear constraint whose boundary does not touch the feasible region is called *strongly redundant.* A redundant linear constraint that can become necessary with the removal of some other redundant constraints is called *relatively redundant*, otherwise it is called *absolutely redundant.* The

LMI constraint $A(x) \succeq 0$ is called redundant, necessary, weakly redundant, strongly redundant, relatively redundant or absolutely redundant with respect to $\mathcal{R}$ in a similar way. An excellent discussion on these concepts is given in [7].

Consider the *kth* linear constraint in (2.2) and the following associated semidefinite programming problem:

$$\begin{aligned} SDP_k: \quad & min \;\; a_k^T x \\ & s.t. \;\; x \in \mathcal{R}_k \end{aligned}$$

The dual of SDP_k is

$$\begin{aligned} DSDP_k: \; max \;\; & -A_0 \bullet Z - \sum_{j=1, j\neq k}^{q} b_j y_j \\ s.t. \;\; & A_i \bullet Z + \sum_{j=1, j\neq k}^{q} (a_j)_i y_j = (a_k)_i \quad (i = 1, 2, \ldots, n), \\ & Z \succeq 0 \\ & y_j \geq 0 \quad (j = 1 \ldots, k-1, k+1, \ldots, q), \end{aligned}$$

where the variables are $m \times m$ symmetric matrix Z and $y_j \in \mathbb{R}$ $(j = 1 \ldots, k-1, k+1, \ldots, q)$. $A \bullet B = \sum_{i,j} a_{ij} b_{ij}$ is the standard inner product on the set of real matrices. We assume that:

Assumption 1: SDP_k is strictly feasible.

Observe that if $\mathcal{R}$ is strictly feasible, so is each SDP_k.

The following theorem gives the main foundation of our methods for identifying redundant linear constraints from the system (2.1) and (2.2).

THEOREM 2.1. *The linear constraint $b_k + a_k^T x \geq 0$ is redundant with respect to $\mathcal{R}$ if and only if the infimum p_k^* of the objective function values of SDP_k exists that satisfies $b_k + p_k^* \geq 0$.*

Proof ($\Rightarrow$) Suppose $b_k + a_k^T x \geq 0$ is redundant. Then $\mathcal{R}_k = \mathcal{R}$, and so for each $x \in \mathcal{R}_k$, we have $b_k + a_k^T x \geq 0$. Hence, the set $\{a_k^T x : x \in \mathcal{R}_k\}$ is bounded from below by $-b_k$.

By Assumption 1, the infimum p_k^* of the objective function values of SDP_k exists that satisfies $p_k^* \geq -b_k$, that is, $b_k + p_k^* \geq 0$. ($\Leftarrow$) Suppose $b_k + a_k^T x \geq 0$ is necessary. Then, there exists a point $\tilde{x}$ such that $\tilde{x} \in \mathcal{R}_k$ but $b_k + a_k^T \tilde{x} < 0$. If the objective values of SDP_k has no infimum, then the proof is done. If the infimum p_k^* of the objective function values of SDP_k exists, then $p_k^* \leq a_k^T \tilde{x}$ and hence $b_k + p_k^* \leq b_k + a_k^T \tilde{x} < 0$. □

It is informative to note that in the above theorem, the optimal solution of SDP_k may not be attained in $\mathcal{R}_k$. Consider the following example from [24]:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + x_1 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \succeq 0$$
$$x_1 \geq 0 \qquad (1)$$

Clearly, $x_1 \geq 0$ is implied by the linear matrix inequality constraint and thus redundant. But, SDP_1 has no optimal solution (x_1^*, x_2^*) in the feasible region with $x_1^* \geq 0$. Here, the dual of SDP_1 is not strictly feasible. If the dual $DSDP_k$ is strictly feasible, then SDP_k attains its optimal solution in $\mathcal{R}_k$ [15]. We assume:

Assumption 2: $DSDP_k$ is strictly feasible.

We remark that if $DSDP_k$ is not strictly feasible, but it has a nonempty relative interior, then an equivalent problem that is strictly feasible can be constructed by projecting the minimal face as described in [15]. We state the following corollary to Theorem 2.1:

COROLLARY 2.1. *The linear constraint $b_k + a_k^T x \geq 0$ is redundant with respect to $\mathcal{R}$ if and only if SDP_k has an optimal solution x_k^* in $\mathcal{R}_k$ that satisfies $b_k + a_k^T x_k^* \geq 0$.*

Proof By Assumption 2, SDP_k attains its optimal solution in $\mathcal{R}_k$. The rest follows from Theorem 2.1. □

Corollary 2.1 shows that all redundant linear constraints can be found by solving SDP_k for $k = 1, 2, \ldots, q$. In the next section, we provide short-cuts less expensive than solving SDP_k for identifying some of the linear constraints as redundant or not. Recall that by Assumption 1, each SDP_k is strictly feasible.

LEMMA 2.1. *([24] A point $x_k^* \in \mathbb{R}^n$ is optimal to SDP_k if and only if there is $Z^* \succeq 0$ and $y_j^* \geq 0$ $(j = 1, \ldots, k-1, k+1, \ldots, q)$ such that the following* KKT *conditions hold:*

$$A(x_k^*) \succeq 0, \ b_j + a_j^T x_k^* \geq 0 \ (j = 1, \ldots, k-1, k+1, \ldots, q) \tag{2.5}$$

$$A_i \bullet Z^* + \textstyle\sum_{j=1, j\neq k}^{q} (a_j)_i y_j^* = (a_k)_i \ (i = 1, 2, \ldots, n) \tag{2.6}$$

$$A(x_k^*) Z^* = 0, \ (b_j + a_j^T x_k^*) y_j^* = 0 \ (j = 1, \ldots, k-1, k+1, \ldots, q) \tag{2.7}$$

PROPOSITION 2.1. *Let $x_k^* \in \mathbb{R}^n$. Let $\operatorname{rank}(A(x_k^*)) = r$ and consider an orthogonal decomposition of $A(x_k^*)$*

$$A(x_k^*) = [Q_1 Q_2] \operatorname{diag}(0, \ldots, 0, \lambda_1, \lambda_2, \ldots, \lambda_r) [Q_1 Q_2]^T.$$

where $\lambda_1, \ldots, \lambda_r$ are the nonzero eigenvalues of $A(x_k^*)$. Then x_k^* is optimal to SDP_k if and only if there is $Z^* \succeq 0$ and $y_j^* \geq 0$ $(j = 1, \ldots, k-1, k+1, \ldots, q)$ such that the following KKT *conditions hold:*

$$A(x_k^*) \succeq 0, \ b_j + a_j^T x_k^* \geq 0 \ (j = 1, \ldots, k-1, k+1, \ldots, q)$$
$$Q_1^T A_i Q_1 \bullet Q_1^T Z^* Q_1 + \textstyle\sum_{j=1, j\neq k}^{q} (a_j)_i y_j^* = (a_k)_i \ (i = 1, 2, \ldots, n)$$
$$A(x_k^*) Z^* = 0, \ (b_j + a_j^T x_k^*) y_j^* = 0 \ (j = 1, \ldots, k-1, k+1, \ldots, q)$$

Proof: By the assumptions, Lemma 2.1 holds. It suffices to show that $A_i \bullet Z^* = Q_1^T A_i Q_1 \bullet Q_1^T Z^* Q_1$ for $(i = 1, 2, \ldots, n)$. We have:

$$\begin{aligned} A(x_k^*) Z^* = 0 &\iff [Q_1 Q_2]^T A(x_k^*) Z^* [Q_1 Q_2] = 0 \\ &\iff \operatorname{diag}(0, \ldots, 0, \lambda_1, \lambda_2, \ldots, \lambda_r) [Q_1 Q_2]^T Z^* [Q_1 Q_2] = 0 \\ &\iff Q_2^T Z^* Q_2 = 0 \text{ and } Q_1^T Z^* Q_2 = 0. \end{aligned}$$

Hence, we obtain

$$\begin{aligned} A_i \bullet Z^* &= [Q_1 Q_2]^T A_i [Q_1 Q_2] \bullet [Q_1 Q_2]^T Z^* [Q_1 Q_2] \\ &= Q_1^T A_i Q_1 \bullet Q_1^T Z^* Q_1 + Q_2^T A_i Q_2 \bullet Q_2^T Z^* Q_2 + Q_2^T A_i Q_1 \bullet Q_2^T Z^* Q_1 + Q_1^T A_i Q_2 \bullet Q_1^T Z^* Q_2 \\ &= Q_1^T A_i Q_1 \bullet Q_1^T Z^* Q_1 \ \square \end{aligned}$$

We will use the following example to illustrate some of the concepts and methods.

Main Example: Consider the following system of linear and LMI constraints. There is one LMI and seven linear constraints. The feasible region is given in Figure 1.

$$\begin{bmatrix} 5 & -1 & 0 & 0 \\ -1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} + x_1 \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + x_2 \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \succeq 0 \quad (8)$$

$37 + x_1 - 14x_2 \geq 0 \quad (1)$

$14 + 3x_1 - 8x_2 \geq 0 \quad (2)$

$5 - x_1 - x_2 \geq 0 \quad (3)$

$3 + x_1 + x_2 \geq 0 \quad (4)$

$6 - x_1 - x_2 \geq 0 \quad (5)$

$11 + 5x_1 + 2x_2 \geq 0 \quad (6)$

$10 - x_1 + 2x_2 \geq 0 \quad (7)$

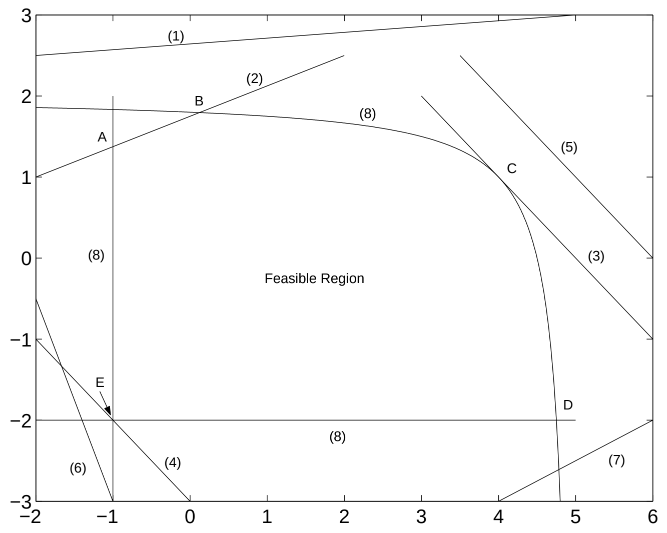

Figure 1. The feasible region of main example

3. Normal Cone at a Boundary Point

In this section, we discuss a useful characterization of the normal cone at a boundary point of an LMI.

Consider the system (2.1) and (2.2) and the corresponding feasible region $\mathcal{R}$. Let x^* be a boundary point of $\mathcal{R}$. The linear constraint $b_j + a_j^T x \geq 0$ (2.2) is said to be *binding* at x^* if $b_j + a_j^T x^* = 0$. The LMI constraint $A(x) \succeq 0$ (2.1) is said to be *binding* at x^* if $A(x^*) \succeq 0$ and $A(x^*) \not\succ 0$. Note that if a constraint is binding at x^*, it means that x^* is a boundary point of the constraint.

Recall that if $b_j + a_j^T x \geq 0$ is binding at x^*, then the normal vector to the linear constraint at x^* is simply a_j. What is the normal vector (or the normal cone) to $A(x) \succeq 0$ at x^* if $A(x) \succeq 0$ is binding at x^*? We discuss this question in the following.

We now assume throughout this section that x^* is a boundary point of the LMI $A(x) \succeq 0$. Let $\text{rank}(A(x^*)) = r$ and consider an orthogonal decomposition of $A(x^*)$

$$A(x^*) = [Q_1 Q_2]\text{diag}(0, \dots, 0, \lambda_1, \lambda_2, \dots, \lambda_r)[Q_1 Q_2]^T.$$

where $\lambda_1, \dots, \lambda_r$ are the positive eigenvalues of $A(x^*)$. Let

$$\mathcal{R}^A = \{x \in \mathbb{R}^n : A(x) \succeq 0\}.$$

Note that since x^* is a boundary point of $\mathcal{R}^A$, then $r < m$ because $A(x^*)$ must have at least one zero eigenvalue. If the region $\mathcal{R}^A$ is nonsmooth at x^*, then the zero eigenvalue of $A(x^*)$ has multiplicity greater than one [13]. We add the following assumption

Assumption 3: The matrices $\{A_i : \ i = 1, \dots, n\}$ are linearly independent.

Let $\mathcal{K}$ be the set of all $m \times m$ positive semidefinite matrices. Consider the set

$$\mathcal{M}_r = \{Z \in \mathcal{K} : \text{rank}(Z) = r\}.$$

Let $\mathcal{S}^r$ the set of all $r \times r$ real symmetric matrices. By [2] the tangent space to $\mathcal{M}_r$ at $A(x^*)$ is given by:

$$T_{A(x^*)} = \{[Q_1 Q_2] \begin{bmatrix} 0 & V \\ V^T & W \end{bmatrix} [Q_1 Q_2]^T : V \in \mathbb{R}^{(m-r)\times r}, W \in \mathcal{S}^r\}$$

Let

$$\Phi = \{Z \succeq 0 : Z = A(x) \text{ for some } x \in \mathbb{R}^n\}$$

Note that $T_{A(x^*)}$ is also the tangent space to Φ at $A(x^*)$. Since the matrices $\{A_i : \ i = 1, \ldots, n\}$ the linearly independent, for each $Z \in \Phi$, there is a unique point x in $\mathbb{R}^n$ for which $Z = A(x)$. The mapping $f : \Phi \mapsto \mathcal{R}^A$ defined by $f(A(x)) = x$, gives a one-to-one correspondence between the sets Φ and $\mathcal{R}^A$.

LEMMA 3.1. *The tangent space to $\mathcal{R}^A$ at x^* is given by*

$$tan(x^*, \mathcal{R}^A) = \{x \in \mathbb{R}^n : A(x) = Z \ for \text{ some Z} \in T_{A(x^*)}\}$$

Proof: We already know that the mapping $f : A(x) \mapsto x$ is one-to-one. It suffices to show that the mapping $f^{-1} : x \mapsto A(x)$ is affine. Observe that $f^{-1}(x) = A_0 + \sum_{i=1}^n x_i A_i$, and so f^{-1} is affine. Hence, $T_{A(x^*)}$ is the tangent space to Φ at $A(x^*)$ if and only if $tan(x^*, \mathcal{R}^A)$ is the tangent space to $\mathcal{R}^A$ at x^* □

COROLLARY 3.1. *The tangent space to $\mathcal{R}^A$ at x^* is given by*

$$tan(x^*, \mathcal{R}^A) = \{x : Q_1^T A_0 Q_1 + \sum_{i=1}^n x_i Q_1^T A_i Q_1 = 0\}$$

Proof: By Lemma 3.1, $tan(x^*, \mathcal{R}^A)$ is the set of points x that satisfy

$$A(x) = [Q_1 Q_2] \begin{bmatrix} 0 & V \\ V^T & W \end{bmatrix} [Q_1 Q_2]^T$$

for some $V \in \mathbb{R}^{(m-r) \times r}, W \in \mathcal{S}^r$. This is equivalent to the system

$$Q_1^T A(x) Q_1 = 0 \Longleftrightarrow Q_1^T A_0 Q_1 + \sum_{i=1}^n x_i Q_1^T A_i Q_1 = 0. \ \square$$

COROLLARY 3.2. *If $\mathcal{R}^A$ is smooth at x^*, then the gradient (inward normal vector) y to $\mathcal{R}^A$ at x^* is given by*

$$y_i = Q_1^T A_i Q_1 \ (i = 1, 2, \ldots, n).$$

Proof: If $\mathcal{R}^A$ is smooth at x^*, then the equation $Q_1^T A_0 Q_1 + \sum_{i=1}^n x_i Q_1^T A_i Q_1 = 0$ of the tangent space is a linear equation. The gradient to $\mathcal{R}^A$ at x^* is the vector $(Q_1^T A_1 Q_1, \ldots, Q_1^T A_n Q_1)^T$ normal to the line. $\Box$

If $\mathcal{R}^A$ is smooth at x^*, it follows from Corollary 3.2, that the outward normal vector (or simply the normal vector) y to $\mathcal{R}^A$ at x^* is given by

$$y_i = -Q_1^T A_i Q_1 \ (i = 1, 2, \ldots, n).$$

The tangent cone to $\mathcal{R}^A$ at x^* is defined by

$$tcone(x^*, \mathcal{R}^A) = Cl\{d \in \mathbb{R}^n : x^* + td \in \mathcal{R}^A \text{ for small } t > 0\}.$$

The normal cone to $\mathcal{R}^A$ at x^* is defined by

$$ncone(x^*, \mathcal{R}^A) = \{d \in \mathbb{R}^n : d^T(z - x^*) \leq 0 \text{ for all } z \in \mathcal{R}^A\}.$$

If $\mathcal{R}^A$ is smooth at x^*, then $ncone(x^*, \mathcal{R}^A)$ is a singleton set containing the normal vector at x^*. For more information on $tcone(x^*, \mathcal{R}^A)$ and $ncone(x^*, \mathcal{R}^A)$, see ([22], [19]).

PROPOSITION 3.1. *Let $X \in \mathcal{S}^{m-r}$ and let d be the vector defined by $d_i = -(Q_1^T A_i Q_1) \bullet X$ $(i = 1, 2, \ldots, n)$. If $X \succeq 0$, then d belongs to $ncone(x^*, \mathcal{R}^A)$.*

Proof: By the definition of Q_1, $Q_1^T A_0 Q_1 + \sum_{i=1}^n x_i^* Q_1^T A_i Q_1 = 0$. Let $z \in \mathcal{R}^A$ and $X \succeq 0$. Then

$$\begin{aligned} d^T(z - x^*) &= -\sum_{i=1}^n ((Q_1^T A_i Q_1) \bullet X)(z_i - x_i^*) \\ &= -(\sum_{i=1}^n z_i Q_1^T A_i Q_1 - \sum_{i=1}^n x_i^* Q_1^T A_i Q_1) \bullet X \\ &= -(\sum_{i=1}^n z_i Q_1^T A_i Q_1 + Q_1^T A_0 Q_1) \bullet X \\ &= -(Q_1^T A(z) Q_1) \bullet X \\ &\leq 0. \ \Box \end{aligned}$$

By [22], $ncone(x^*, \mathcal{R}^A)$ is the negative dual of $tcone(x^*, \mathcal{R}^A)$. This and [19] give the following result.

LEMMA 3.2.

$$ncone(x^*, \mathcal{R}^A) = Cl\{d \in \mathbb{R}^n : d_i = -(Q_1^T A_i Q_1) \bullet X \ (i = 1, 2, \ldots, n) \text{ for } X \succeq 0\}. \ \square$$

It follows from Lemma 3.2 that if the set $\{d \in \mathbb{R}^n : d_i = -(Q_1^T A_i Q_1) \bullet X \ (i = 1, 2, \ldots, n) \text{ for } X \succeq 0\}$ is closed, then

$$ncone(x^*, \mathcal{R}^A) = \{d \in \mathbb{R}^n : d_i = -(Q_1^T A_i Q_1) \bullet X \ (i = 1, 2, \ldots, n) \text{ for } X \succeq 0\}.$$

Example 1: Consider the point $D = (4.75, -2)^T$ on the boundary of LMI (8). We have that

$$Q_1 = \begin{bmatrix} -0.9701 & 0 \\ -0.2425 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

Hence, the normal cone to LMI (8) at D is

$$Cl\left\{ \begin{pmatrix} 0.9412x_{11} \\ 0.0588x_{11} - x_{22} \end{pmatrix} : x_{11} \geq 0, \ x_{22} \geq 0 \right\} \ \square$$

4. Identification at Smooth Boundary Point of $\mathcal{R}^A$

We present short-cuts for identifying certain redundant linear constraints that are binding at a smooth boundary point of $\mathcal{R}^A$.

Suppose constraint k is redundant and let x_k^* be an optimal solution of SDP_k. The following propositions provide a quick method for determining redundancy of some of the other linear constraints binding at x_k^* after the solution of SDP_k is found. We assume that SDP_k is strictly feasible.

PROPOSITION 4.1. *Suppose either of the following holds*

1. *$A(x) \succeq 0$ is binding and smooth at x_k^**

2. *$A(x) \succeq 0$ is not binding at x_k^*.*

If the gradients of all the constraints binding at x_k^* are linearly independent, then such constraints are necessary.

Proof: Note that if $A(x) \succeq 0$ is binding and smooth at x_k^*, then the gradient of $A(x) \succeq 0$ at x_k^* is defined (Corollary 3.2). Consider the set of all constraints binding at x_k^*. By linear independence of the gradients, if any of the binding constraints is removed, then the feasible region $\mathcal{R}$ will change. Hence, the binding constraints are necessary. □

The following example illustrates Proposition 4.1.

Example 2: Consider SDP_1 corresponding to linear constraint (1). Its optimal solution is $x_1^* = B = (0.1202, 1.7951)^T$. This satisfies $b_1 + a_1^T x_1^* \geq 0$. So, (1) is redundant. Consider constraints (2) and (8) that are binding at x_1^*. The gradient of (2) at x_1^* is $(3, -8)^T$. To find the gradient to (8) at x_1^*, we obtain $Q_1 = (-0.2008, -0.9796, 0, 0)^T$. So, the gradient $(Q_1^T A_1 Q_1, Q_1^T A_2 Q_1)^T = (-0.0403, -0.9597)^T$. Since the two gradients are linearly independent, then by Proposition 4.1, constraints (2) and (8) are necessary. □

Let I_k be the index set of all linear constraints binding at x_k^*. Suppose $t \in I_k$.

LEMMA 4.1. *The point x_k^* is optimal to SDP_t if and only if there is $Z^* \succeq 0$ and $y_j^* \geq 0$ $(j = 1, \ldots, t-1, t+1, \ldots, q)$ such that the following* KKT *conditions hold:*

$$A(x_k^*) \succeq 0,\ b_j + a_j^T x_k^* \geq 0 \text{ for } j = 1, \ldots, t-1, t+1, \ldots, q \tag{4.1}$$

$$Q_1^T A_i Q_1 \bullet Q_1^T Z^* Q_1 + \textstyle\sum_{j=1, j\neq t}^{q} (a_j)_i y_j^* = (a_t)_i \ (i = 1, 2, \ldots, n) \tag{4.2}$$

$$A(x_k^*) Z^* = 0,\ (b_j + a_j^T x_k^*) y_j^* = 0 \text{ for } j = 1, \ldots, t-1, t+1, \ldots, q \tag{4.3}$$

Proof: This follows directly from Proposition 2.1 that corresponds to SDP_t. □

THEOREM 4.1. *Suppose that either of the following holds*

1. *$A(x) \succeq 0$ is binding and smooth at x_k^* and the zero eigenvalue of $A(x_k^*)$ has multiplicity one.*

2. *$A(x) \succeq 0$ is not binding at x_k^*.*

Then t is redundant if and only if the following system is feasible:

$$\alpha Q_1^T A_i Q_1 + \sum_{j \in I_k \setminus \{t\}} u_j (a_j)_i \ = \ (a_t)_i \ (i = 1, 2, \ldots, n) \tag{4.4}$$

$$\alpha, u_j \ \geq \ 0 \text{ for } j \in I_k \setminus \{t\} \tag{4.5}$$

where we take $\alpha = 0$ for the case when $A(x) \succeq 0$ is not binding at x_k^*.

Proof: We proof the case when $A(x) \succeq 0$ is binding and smooth at x_k^* and the zero eigenvalue of $A(x_k^*)$ has multplicity one. The proof of the other case is given in [18]. Now, Q_1 is of dimension $n \times 1$. Suppose the system (4.4)-(4.5) is feasible and consider a feasible solution α, u_j for $j \in I_k \setminus \{t\}$. Let $Z^* = \alpha Q_1 Q_1^T$. Then

$$\alpha[Q_1^T A_i Q_1] = \alpha[(Q_1^T A_i Q_1) \bullet (Q_1^T (Q_1 Q_1^T) Q_1)] = (Q_1^T A_i Q_1) \bullet (Q_1^T Z^* Q_1).$$

Also, $A(x_k^*)Z^*{=}0$. Let $u_j = 0$ for $j \notin I_k \setminus \{t\}$ and $j \neq t$. Then, the optimality conditions (4.1)-(4.3) of SDP_t hold with x_k^*, $Z^* = \alpha Q_1 Q_1^T$ and $y_j^* = u_j$ for $j = 1, \ldots, t-1, t+1, \ldots, q$. It follows by Lemma 4.1 that x_k^* is an optimal solution of SDP_t and $b_t + a_t^T x_k^* = 0$. Hence, by Corollary 2.1, t is redundant. Conversely, suppose t is redundant. Since $t \in I_k$, that is, $b_t + a_t^T x \geq 0$ is binding at x_k^*, then x_k^* is an optimal solution of SDP_t. Hence, by Lemma 4.1, the optimality conditions (4.1)-(4.3) must hold for some $Z^* \succeq 0$ and $y_j^* \geq 0$ $j = 1, \ldots, t-1, t+1, \ldots, q$. Note that if $X \succeq 0$, then by Corollary 3.2 and Lemma 3.2

$$(Q_1^T A_i Q_1) \bullet X \quad (i = 1, 2, \ldots, n).$$

is either zero or a nonzero multiple of the gradient of $A(x) \succeq 0$ at x_k^*. So, in the optimality conditions (4.2), for each i

$$Q_1^T A_i Q_1 \bullet Q_1^T Z^* Q_1 = \alpha^* [Q_1^T A_i Q_1 \bullet Q_1^T I_m Q_1] = \alpha^* Q_1^T A_i Q_1$$

for some $\alpha^* \geq 0$. Also, by (4.3), $y_j^* = 0$ for $j \notin I_k \setminus \{t\}$. It follows that the system (4.4)-(4.5) is feasible with α^*, Z^* and $u_j = y_j^*$ for $j = 1, \ldots, t-1, t+1, \ldots, q$. $\square$

If the system (4.4)-(4.5) is feasible, then the vector a_t is a nonnegative linear combination of the gradients of the other constraints binding at x_k^*. Note that this system is a simple linear feasibility problem that uses only those constraints binding at x_k^*. It is easier to solve than solving SDP_t itself to determine the redundancy of t.

5. Identification at Nonsmooth Boundary Point of $\mathcal{R}^A$

In this section, we consider the case when $A(x) \succeq 0$ is binding at x_k^* such that the region $\mathcal{R}^A$ is nonsmooth at x_k^*. As in the previous section, we develop a short-cut for determining the redundancy of other linear constraints binding at x_k^* after SDP_k is solved.

Suppose $A(x) \succeq 0$ is binding at x_k^* and $\mathcal{R}^A$ is nonsmooth at x_k^*. Let I_k be the index set of all linear constraints binding at x_k^*.

THEOREM 5.1. *Constraint t is redundant if and only if the following system is feasible:*

$$(Q_1^T A_i Q_1) \bullet (Q_1^T Z Q_1) + \sum_{j \in I_k \setminus \{t\}} u_j (a_j)_i = (a_t)_i \ (i = 1, 2, \ldots, n) \tag{5.1}$$

$$Z \succeq 0 \tag{5.2}$$

$$u_j \geq 0 \text{ for } j \in I_k \setminus \{t\} \tag{5.3}$$

Proof: The proof is analogous to that of the first case of Theorem 4.1. □

Note that system (5.1)-(5.3) is an SDP feasibility problem that is easier to solve than SDP_k. The system only uses those constraints that are binding at x_k^*. Recall from Lemma 3.2 that the normal cone to $\mathcal{R}^A$ at x_k^* is given by

$$ncone(x_k^*, \mathcal{R}^A) = Cl\{d \in \mathbb{R}^n : d_i = -Q_1^T A_i Q_1) \bullet X \ (i = 1, 2, \ldots, n) \text{ for } X \succeq 0\}.$$

Hence, if the system (5.1)-(5.3) is feasible, then the vector $-a_t$ is a nonnegative linear combination of a vector in $ncone(x_k^*, \mathcal{R}^A)$ and the negative gradients of the other linear constraints binding at x_k^*. To illustrate Theorem 5.1, we give the following simple example:

Example 3: Consider SDP_6. The optimal solution is at $x_6^* = E = (-1, -2)^T$. Since x_6^* satisfies $b_6 + a_6^T x_6^* \geq 0$, then (6) is redundant. Consider the linear constraint (4) and LMI (8) that are binding at x_6^*. To determine the redundancy of (4), we consider the corresponding system (5.1)-(5.3). Note that

$$Q_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The system is:

$$z_{33} = 1 \;\; z_{44} = 1, \;\; z_{33} \geq 0, \;\; z_{44} \geq 0.$$

Since the system is feasible, then by Theorem 5.1, (4) is redundant. Here, the feasibility of the system means that the negative of the vector $a_4 = (1, 1)^T$ lies in the normal cone to LMI (8) at E. $\square$

The results presented in this section and in the previous sections describe our methods for finding all redundant linear constraints from the system (2.1)-(2.2). These can be used to give an algorithm for finding all redundant linear constraints from the system.

6. Conclusion

We have studied the problem of identifying all redundant linear constraints from a system of linear matrix inequality constraints. The problem is of great interest in semidefinite programming with regards to computational and decision issues. We presented deterministic methods for finding all redundant linear constraints based on solving certain semidefinite programming problems (SDP). We also gave short-cuts for determining redundancy of linear constraints that are binding at an optimal point. These short-cuts each requires solving a simple SDP feasibility problem.

Our methods extend some of those from linear programming to semidefinite programming. While the cost of identifying all redundant linear constraints can sometimes be expensive, there are possible benefits to the knowledge obtained. This topic is of interest in its own right. We hope this work will add interest to the problem of redundancy in semidefinite programming. This research is preliminary and we hope in the future to investigate how our methods would actually work in practice. Finally, we note that the LMI considered in the problem formulation can have redundant linear constraints of its own. It would be useful to know how to identify all such redundancy. This would require further research.

Acknowledgements

This project was partially supported by an *REU-NSF* grant of Northern Arizona University. The authors would like to thank James W. Swift, Lieven Vandenberghe and Brian Borchers for responding to our questions and for their suggestions.

References

1. F. Alizadeh, J.A. Haeberly, M. V. Nayakkankuppam, M. L. Overton and S. Schmieta, "*SDPpack - Version 0.9 Beta for Matlab 5.0*", User's Guide, 1997.
2. F. Alizadeh, J.A. Haeberly and M. Overton, "Complementarity and Nondegeneracy in Semidefinite Programming", *Math. Programming*, vol. 77, no. 2, ser. B, (1997) 111-128.
3. F. Alizadeh, "Interior Point Methods in Semidefinite Programming with Applications to Combinatorial Optimization", *SIAM Journal on Optimization*, vol. 5, no. 1, (1995) 13-51.
4. M.L. Balinsky, "An algorithm for Finding all Vertices of Convex Polyhedron Sets", *Journal of the Society for Industrial and Applied Mathematics*, vol. 9, no. 1, (1961) 72-88.
5. S.J. Benson, Y. Ye and X. Zhang, "Solving Large-Scale Sparse Semidefinite Programs for Combinatorial Optimization", *SIAM Journal on Optimization*, vol. 10, no. 2, (2000) 443-461.
6. A. Boneh and A. Golan, "Constraint Redundancy and feasible region boundedness by a random feasible point generator (RFPG), contributed paper", EURO III, Amsterdam, 1979.
7. A. Boneh, S. Boneh and R.J. Caron, "Constraint Classification in Mathematical Programming", *Math. Programming*, vol. 61, (1993) 61-73.
8. B. Borchers, "CSDP, a C library for semidefinite programming", *Optimization Methods & Software*, vol. 11, no.1, (1999) 613-23
9. B. Borchers and J. Young, "How Far Can We Go With Primal-Dual Interior Point Methods for SDP?", *Optimization online, http://www.optimization-online.org/DB_FILE/2005/03/1082.pdf*, 2005
10. S. Boyd and L. El Ghaoui, "Linear Matrix Inequalities in System and Control Theory", *SIAM*, vol. 15, Philadelphia, PA, 1994.
11. R.J. Caron, A. Boneh, S. Boneh, "Redundancy", in: *Advances in Sensitivity Analysis and Parametric Programming*, Kluwer Academic Publishers, International Series in Operations Research and Management Science, vol. 6, 1997.

12. R.J. Caron, “A Degenerate Extreme Point Strategy for the Classification of Linear Constraints as Redundant or Necessary”, *Journal of Optimization Theory and Applications*, vol. 62, (1989) 225-237.

13. J. Cullum, W.E. Donath and P. Wolfe, “The Minimization of Certain Nondifferentiable Sums of Eigenvalue Problems”, *Mathematical Programming*, vol. 3, (1975) 35-55.

14. G. Gruber, S. Kruk, F. Rendl and H. Wolkowicz,“Presolving for Semidefinite Programs Without Constraint Qualifications”, Technical Report CORR 98-32, University of Waterloo, Waterloo, Ontario, 1998.

15. C. Helmberg, “*Semidefinite Programming for Combinatorial Optimization*”, Technical Report ZR-00-34, TU Berlin, Konrad-Zuse-Zentrum, Berlin, 2000.

16. S. Jibrin and I.S. Pressman, “Monte Carlo Algorithms for the Detection of Necessary Linear Matrix Inequality Constraints”, *International Journal of Nonlinear Sciences and Numerical Simulation*, vol. 2, (2001) 139-154.

17. S. Jibrin, “Redundancy in Semidefinite Programming”, Ph. D Thesis, Carleton University, Canada, 1998.

18. M. H. Karwan, V. Lotfi, J. Telgen and S. Zionts, “*Redundancy in Mathematical Programming: A State-of-the-Art Survey*”, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1983

19. K. Krishnan and J.E. Mitchell, “Properties of a Cutting Plane Method for Semidefinite Programming”, *Optimization online, http://www.optimization-online.org/DB_HTML/2003/05/655.html*, 2003

20. I.J. Lustig, R.E. Marsten and D.F. Shanno, “Interior Point Methods for Linear Programming: Computatioal State of the Art”, *ORSA Journal on Computing*, vol. 6, (1994) 1-14.

21. Yu. Nesterov and A. Nemirovsky, “*Interior-point polynomial methods in convex programming*”, volume 13 of Studies in Applied Mathematics, SIAM, Philadelphia, PA, 1994.

22. G. Pataki, “The Geometry of Semidefinite Programming”, in: H. Wolkowicz, R. Saigal, L. Vandenberghe eds, *Handbook of Semidefinite Programming: Theory, Algorithms and Applications*, Kluwer’s International Series, (2000) 29-64

23. J. F. Sturm, “Using sedumi 1.02, a matlab toolbox for optimizing over symmetric cones”, *Optimization Methods & Software*, vol. 11-12, (1999) 625-653

24. L. Vandenberghe and S. Boyd, “Semidefinite Programming”, *SIAM Review*, vol. 31, no. 1, (1996) 49-95.

25. L. Vandenberghe and S. Boyd, "Primal-dual potential reduction method for problems involving matrix inequalities", *Math. Programming*, vol. 69 , (1995) 205-236.