Solving Conic Systems via Projection and Rescaling

Javier Peña* Negar Soheili†

December 26, 2016

Abstract

We propose a simple *projection and rescaling algorithm* to solve the feasibility problem

$$\text{find } x \in L \cap \Omega,$$

where L and Ω are respectively a linear subspace and the interior of a symmetric cone in a finite-dimensional vector space V.

This projection and rescaling algorithm is inspired by previous work on rescaled versions of the perceptron algorithm and by Chubanov's projection-based method for linear feasibility problems. As in these predecessors, each main iteration of our algorithm contains two steps: a *basic procedure* and a *rescaling* step. When $L \cap \Omega \neq \emptyset$, the projection and rescaling algorithm finds a point $x \in L \cap \Omega$ in at most $\mathcal{O}(\log(1/\delta(L \cap \Omega)))$ iterations, where $\delta(L \cap \Omega) \in (0, 1]$ is a measure of the most interior point in $L \cap \Omega$. The ideal value $\delta(L \cap \Omega) = 1$ is attained when $L \cap \Omega$ contains the center of the symmetric cone Ω.

We describe several possible implementations for the basic procedure including a perceptron scheme and a smooth perceptron scheme. The perceptron scheme requires $\mathcal{O}(r^4)$ perceptron updates and the smooth perceptron scheme requires $\mathcal{O}(r^2)$ smooth perceptron updates, where r stands for the Jordan algebra rank of V.

1 Introduction

We propose a simple algorithm based on projection and rescaling operations to solve the feasibility problem

$$\text{find } x \in L \cap \Omega, \tag{1}$$

where L and Ω are respectively a linear subspace and the interior of a symmetric cone in a finite-dimensional vector space V. Problem (1) is fundamental in optimization as it encompasses a large class of feasibility problems. For example, for $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, the problem $Ax = b, x > 0$ can be formulated as (1) by taking $L = \{(x, t) \in \mathbb{R}^{n+1} : Ax - tb = 0\}$ and $\Omega = \mathbb{R}^{n+1}_{++}$. For $A \in \mathbb{R}^{m \times n}$, $c \in \mathbb{R}^n$, the problem $A^{\mathrm{T}} y < c$

*Tepper School of Business, Carnegie Mellon University, USA, jfp@andrew.cmu.edu

†College of Business Administration, University of Illinois at Chicago, USA, nazad@uic.edu

can be formulated as (1) by taking $L = \{(s,t) \in \mathbb{R}^{n+1} : tc - s \in \text{span}(A)\}$ and $\Omega = \mathbb{R}^{n+1}_{++}$. Likewise, the strict semi-definite feasibility problem $AX = b, X \in \mathbb{S}^n_{++}$ can be formulated as (1) by taking $L = \{(X,t) \in \mathbb{S}^n \times \mathbb{R} : AX - tb = 0\}$ and $\Omega = \mathbb{S}^n_{++} \times \mathbb{R}_{++}$. The problem of finding an ϵ-solution to a primal-dual pair of conic optimization problems satisfying the Slater condition can also be recast as a problem of the form (1) via a similar type of homogenization.

To solve (1), we consider the equivalent problem

$$\text{find } z \in V \text{ such that } P_L z \in \Omega, \tag{2}$$

where $P_L : V \to V$ denotes the orthogonal projection onto the subspace L. Observe that if z is a solution to (2), then $x = P_L z$ is a solution to (1). Conversely, if x is a solution to (1), then $z = x$ is a solution to (2).

Our *projection and rescaling algorithm* for (2) formalizes the following two intuitive ideas. First, if the set $L \cap \Omega$ is *well-conditioned* in the sense that the subspace L contains points well in the interior of Ω, then a *basic procedure,* which relies only on the projection mapping P_L, can easily find a point in $L \cap \Omega$. Second, when the basic procedure does not find a point in $L \cap \Omega$ after some amount of work, information about the problem instance can be inferred so that some type of *rescaling step* can be applied to obtain a better conditioned problem. This two-step procedure eventually terminates with a feasible point in $L \cap \Omega$ provided this set is nonempty.

Our projection and rescaling algorithm is inspired by previous work on rescaled versions of the perceptron algorithm [6, 10, 20] as well as by Chubanov's work on a projection-based algorithm for linear feasibility problems [8]. In particular, the article [20] is concerned with a feasibility problem of the form

$$\text{find } y \in F, \tag{3}$$

where $F \subseteq W$ is an open convex cone in a finite dimensional vector space W, and it is only assumed that a separation oracle for F is available. The gist of the approach in [20] is to enhance a simple relaxation-type algorithm for (3), namely the perceptron method, with a periodic rescaling of the ambient space W. When the set F is *well-conditioned* in the sense that the volume of $F \cap \{y \in \mathbb{R}^m : \|y\|_2 = 1\}$ exceeds a certain minimum threshold, the perceptron algorithm can easily find a point in F. When that is not the case, the perceptron algorithm identifies a direction d in the ambient space W such that a dilation along d increases the volume of $F \cap \{y \in \mathbb{R}^m : \|y\|_2 = 1\}$ by a constant factor. We note that the article [20] was preceded and inspired by the work of Dunagan and Vempala [10] and Belloni, Freund, and Vempala [6], who introduced random rescaling as a technique for enhancing the perceptron algorithm.

Our projection and rescaling algorithm can be seen as an extension of the recent work of Chubanov [8] for the feasibility problem

$$\text{find } \ x > 0 \text{ such that } Ax = 0, \tag{4}$$

where $A \in \mathbb{R}^{m\times n}$. Observe that (4) is a special case of (1) for $L = \ker(A)$ and $\Omega = \mathbb{R}^n_{++}$. Chubanov [8] relies on the equivalent problem

$$\text{find } z \in \mathbb{R}^n \text{ such that } P_L z > 0, \tag{5}$$

where P_L denotes the orthogonal projection onto $L = \ker(A)$. Chubanov [8] proposes an algorithm that combines a *basic procedure* (a relaxation-type algorithm) for (5) with a periodic rescaling of the ambient space $\mathbb{R}^n$. When the set $\{x > 0 : Ax = 0\}$ is *well-conditioned* in the sense that there exists a point in $\{x > 0 : Ax = 0, \|x\|_\infty = 1\}$ whose coordinates are bounded away from zero (for example when $\{x : Ax = 0, \frac{\mathbf{1}}{2} \leq x \leq \mathbf{1}\} \neq \emptyset$) the basic procedure easily finds a solution to (5). When the basic procedure does not easily find a solution, it identifies a coordinate i such that every point in $\{x > 0 : Ax = 0, \|x\|_\infty = 1\}$ satisfies $x_i < 1/2$. Hence a dilation of the ambient space $\mathbb{R}^n$ along the i-th coordinate transforms the set $\{x > 0 : Ax = 0\}$ into a set that is better conditioned. Chubanov shows that when A has rational entries, the resulting algorithm either finds a solution to (4) or concludes that (4) is infeasible in a total number of operations that is polynomial in the bit-length representation of A. The article by Chubanov [8] is similar in spirit to his previous article [7]. Like [20] and its predecessors [6, 10], both [8] and [7], as well as this paper can be seen as enhancements of the classical relaxation method [1, 16]. Chubanov's work has also been revisited and extended by various sets of authors [4, 15, 21]. The numerical experiments reported in the articles by Roos [21] and by Li, Roos, and Terlaky [15] provide promising evidence of the computational effectiveness of Chubanov's method [8] and related variants.

In a similar fashion to the approaches in [8] and in [20], we propose an algorithm for (2) that combines a simple basic procedure with a periodic rescaling of V. The analysis of our approach relies on a suitable *condition measure* $\delta(L \cap \Omega) \in (0, 1]$ associated to the *most interior* point in $L\cap\Omega$. The ideal value $\delta(L\cap\Omega) = 1$ is attained when $L \cap \Omega$ contains the center of the cone Ω. The main steps in our projection and rescaling algorithm can be sketched as follows. When $\delta(L \cap \Omega)$ exceeds a certain threshold, a *basic procedure* easily finds a point $z \in L \cap \Omega$. On the other hand, when that is not the case, the basic procedure identifies a linear automorphism $D : V \to V$ that leaves Ω unchanged and such that $\delta(D(L) \cap \Omega) > 1.5 \cdot \delta(L \cap \Omega)$. The algorithm then continues with the transformed problem

$$\text{find } \ x \in D(L) \cap \Omega.$$

As Theorem 3 below formally shows, if $L \cap \Omega \neq \emptyset$ then the projection and rescaling algorithm finds a point in $L \cap \Omega$ after $\mathcal{O}(\log(1/\delta(L \cap \Omega))$ rounds of this combination of basic procedure and rescaling step.

We describe several *elementary* implementations for the basic procedure including a perceptron scheme [6, 22], a von Neumann scheme [12], and variations of each of them, namely a von Neumann scheme with away steps [18], and a smooth perceptron scheme [25, 26]. A common attractive feature of all of these schemes is their low

computational work per iteration. We show that the first three schemes require $\mathcal{O}(r^4)$ simple updates and the smooth perceptron algorithm requires $\mathcal{O}(r^2)$ simple updates, where r is the Jordan algebra rank of V. In the special case $\Omega = \mathbb{R}^n_{++}$, we have $r = n$ but the first three schemes require $\mathcal{O}(n^3)$ simple updates and the smooth perceptron scheme requires $\mathcal{O}(n^{3/2})$ simple updates.

It is worth noting that the problems (1) and (3) are alternative systems when $F = \{y : A^*y \in \Omega^*\}$ for a linear mapping $A : V \to W$ with $L = \ker(A)$. In this case the rescaling operation in [20] can be seen as a type of *left reconditioning* that transforms A to DA for some isomorphism $D : W \to W$. On the other hand, the rescaling operation in [8] and its general version in this paper can be seen as a type of *right reconditioning* that transforms A to AD for some isomorphism $D : V \to V$ that satisfies $D(\Omega) = \Omega$. These kinds of left and right reconditioning operations are in the same spirit as the left and right preconditioners operations introduced and discussed in [19].

Observe that the reformulations (2) and (5) are amenable to the algorithmic scheme developed in [20] since they are of the form (3). However, the algorithmic scheme in [20] relies solely on separation and hence does not take advantage of the properties of the symmetric cone Ω. Not surprisingly, the algorithmic scheme in [20] applied to (2) could be weaker than the one presented in this paper. In particular, the iteration bound for the *perceptron phase* in the algorithmic scheme in [20] applied to (2) depends on the dimension of the vector space V. By contrast, the iteration bound for the *basic procedure* of the algorithm in this paper depends on the Jordan algebra rank of V which is at most equal to the dimension of V but could be quite a bit smaller. For instance, the Jordan algebra rank of $\mathbb{S}^n$ is n whereas its dimension is $n(n+1)/2$. If V is endowed with the Jordan algebra associated to the second-order cone, then its Jordan algebra rank is only 2 regardless of its dimension.

The main sections of the paper are organized as follows. In Section 2 we describe a Projection and Rescaling Algorithm that is nearly identical to that proposed by Chubanov [8] for the special case of problem (1) when $V = \mathbb{R}^n$ and $\Omega = \mathbb{R}^n_{++}$, albeit presented in a slightly different format. The main purpose of this section is to introduce the algorithmic scheme and main ideas that we subsequently generalize. In Section 3 we extend our Projection and Rescaling Algorithm to the case when V is the space $\mathbb{S}^n$ of symmetric $n \times n$ real matrices and Ω is the cone $\mathbb{S}^n_{++}$ of positive definite matrices. This is a special but particularly important case of the more general case when V is a vector space endowed with an Euclidean Jordan algebra structure and Ω is the interior of the cone of squares in V, which is presented in Section 4. In Section 5 we describe different implementations for the basic procedure. Finally in Section 6 we discuss how the projection matrix in (2) can be updated after each rescaling operation.

2 Projection and rescaling algorithm

Assume $L \subseteq \mathbb{R}^n$ and consider the problem

$$\text{find } x \in L \cap \mathbb{R}^n_{++}. \tag{6}$$

Let $P_L : \mathbb{R}^n \to \mathbb{R}^n$ be the orthogonal projection onto L. Then (6) is equivalent to

$$\text{find } y \in \mathbb{R}^n \text{ such that } P_L y \in \mathbb{R}^n_{++}.$$

Consider the following kind of *condition measure* of the set $L \cap \mathbb{R}^n_{++}$:

$$\delta(L \cap \mathbb{R}^n_{++}) := \max_x \left\{ \prod_{i=1}^n x_i : x \in L \cap \mathbb{R}^n_{++}, \|x\|_2^2 = n \right\}.$$

Observe that $\delta(L \cap \mathbb{R}^n_{++}) > 0$ provided $L \cap \mathbb{R}^n_{++} \neq \emptyset$. Furthermore, by the arithmetic-geometric inequality, $\delta(L \cap \mathbb{R}^n_{++}) \leq 1$, and $\delta(L \cap \mathbb{R}^n_{++}) = 1$ precisely when $e = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^{\mathrm{T}} \in L \cap \mathbb{R}^n_{++}$.

For $v \in \mathbb{R}^n$ let v^+ denote $\max(v, 0)$ componentwise. Let $e_i \in \mathbb{R}^n$ denote the unitary vector whose i-th entry is equal to one and all others are equal to zero. The following key observation suggests a certain rescaling as a reconditioning operation.

Proposition 1 *Assume $z \in \mathbb{R}^n_+ \setminus \{0\}$ is such that $\|(P_L z)^+\|_2 \leq \epsilon \|z\|_\infty$ for some $\epsilon \in (0, 1)$. Let $D = I + a e_i e_i^{\mathrm{T}}$ where i is such that $z_i = \max_{j=1,\ldots,n} z_j$ and $a > 0$. Then*

$$\delta(DL \cap \mathbb{R}^n_{++}) \geq \frac{1+a}{(1+(2a+a^2)\epsilon^2)^{n/2}} \cdot \delta(L \cap \mathbb{R}^n_{++}).$$

In particular, if $\epsilon = \frac{1}{3\sqrt{n}}$ and $a = 1$ then

$$\delta(DL \cap \mathbb{R}^n_{++}) \geq 1.5 \cdot \delta(L \cap \mathbb{R}^n_{++}).$$

Proof: Observe that for $x \in L \cap \mathbb{R}^n_{++}$ the point $\hat{x} := \frac{\sqrt{n}}{\|Dx\|_2} Dx$ satisfies $\hat{x} \in DL \cap \mathbb{R}^n_{++}$ and $\|\hat{x}\|_2^2 = n$. Thus it suffices to show that for $x \in L \cap \mathbb{R}^n_{++}$ with $\|x\|_2^2 = n$ both $\prod_{j=1}^n (Dx)_j = (1+a) \prod_{j=1}^n x_j$ and $\|Dx\|_2^2 \leq n(1 + (2a + a^2)\epsilon^2)$ as this would imply $\prod_{j=1}^n (\hat{x})_j \geq \frac{1+a}{(1+(2a+a^2)\epsilon^2)^{n/2}} \cdot \prod_{j=1}^n x_j$. Equivalently, it suffices to show that for $x \in L \cap \mathbb{R}^n_{++}$ with $\|x\|_2 = 1$ we have $\prod_{j=1}^n (Dx)_j \geq (1+a) \prod_{j=1}^n x_j$ and $\|Dx\|_2^2 \leq 1 + (2a + a^2)\epsilon^2$.

Assume $x \in L \cap \mathbb{R}^n_{++}$ with $\|x\|_2 = 1$ is fixed. Since $Dx = (I + a e_i e_i^{\mathrm{T}})x = x + a x_i e_i$, we have

$$\prod_{j=1}^n (Dx)_j = (1+a) \prod_{j=1}^n x_j.$$

Furthermore, since $x \in L \cap \mathbb{R}^n_{++}$, $\|x\|_2 = 1$, and $z \geq 0$ it follows that

$$0 < x_i z_i \leq x^{\mathrm{T}} z = x^{\mathrm{T}} P_L z \leq \| (P_L z)^+ \|_2 \leq \epsilon z_i.$$

Hence $x_i \leq \epsilon$ and so $\|Dx\|_2^2 = \|x\|_2^2 + (2a + a^2)x_i^2 \leq 1 + (2a + a^2)\epsilon^2$. ■

Proposition 1 suggests the Projection and Rescaling Algorithm described in Algorithm 1 below. We note that Algorithm 1 is nearly identical to the algorithm proposed by Chubanov [8], albeit presented in a slightly different format.

Algorithm 1 Projection and Rescaling Algorithm

1 (**Initialization**)
Let $P_L \in \mathbb{R}^{n \times n}$ be the orthogonal projection onto L.
Let $D := I$ and $P := P_L$.
2 (**Basic Procedure**)
Find $z \gneqq 0$ such that either $Pz > 0$ or $\|(Pz)^+\|_2 \leq \frac{1}{3\sqrt{n}} \|z\|_\infty$.
3 **if** $Pz > 0$ **then** HALT and **return** $x = D^{-1}Pz \in L \cap \mathbb{R}^n_{++}$ **end if**
4 (**Rescaling step**)
Pick i such that $z_i = \|z\|_\infty$.
Put $D := (I + e_i e_i^{\mathrm{T}})D$ and $P := P_{DL}$.
Go back to step 2.

Theorem 1 states the main property of the above algorithm. A major difference from the results in [8] is that Theorem 1 depends solely on $\delta(L \cap \mathbb{R}^n_{++})$. In particular, L can be any arbitrary linear subspace of $\mathbb{R}^n$. It is not necessarily assumed to be the null space of a matrix with rational entries.

Theorem 1 *If $L \cap \mathbb{R}^n_{++} \neq \emptyset$ then Algorithm 1 finds $x \in L \cap \mathbb{R}^n_{++}$ in at most $\log_{1.5}(1/\delta(L \cap \mathbb{R}^n_{++}))$ main iterations.*

Proof: This is an immediate consequence of Proposition 1 and the fact that $\delta(\tilde{L} \cap \mathbb{R}^n_{++}) \leq 1$ for any linear subspace $\tilde{L} \subseteq \mathbb{R}^n$ with $\tilde{L} \cap \mathbb{R}^n_{++} \neq \emptyset$. ■

To complement the statement of Theorem 1, we next account for the number of arithmetic operations required by Algorithm 1. A call to the basic procedure is the bulk of the computational work in each main iteration of Algorithm 1. As we discuss in detail in Section 5, there are several possible implementations for the basic procedure. The simplest implementations for the basic procedure terminate $\mathcal{O}(n^3)$ perceptron or von Neumann steps. Each of these steps requires a matrix-vector multiplication of the form $z \mapsto Pz$ in addition to some other negligible operations. As we explain in Section 6 below, the projection matrix P can be stored and updated in the form $P = QQ^{\mathrm{T}}$ for some matrix $Q \in \mathbb{R}^{n \times m}$ where $m = \dim(L)$ and the columns

of Q form an orthogonal basis of DL. For a matrix of this form, each matrix-vector multiplication $z \mapsto Pz$ requires $\mathcal{O}(mn)$ arithmetic operations. It thus follows that the total number of arithmetic operations required by Algorithm 1 is bounded above by

$$\mathcal{O}(mn \cdot n^3 \cdot \log(1/\delta(L \cap \mathbb{R}^n_{++}))) = \mathcal{O}(mn^4 \log(1/\delta(L \cap \mathbb{R}^n_{++}))).$$

Algorithm 1 is designed to find a solution to (6) assuming that $L \cap \mathbb{R}^n_{++} \neq \emptyset$. If $L \cap \mathbb{R}^n_{++} = \emptyset$, Algorithm 1 will not terminate. However, Algorithm 1 has the straightforward extension described as Algorithm 2 that solves either (6) or its strict alternative

$$\text{find } \hat{x} \in L^\perp \cap \mathbb{R}^n_{++}$$

provided at least one of them is feasible. An immediate consequence of Theorem 1 is that Algorithm 2 will find either $x \in L \cap \mathbb{R}^n_{++}$ or $\hat{x} \in L^\perp \cap \mathbb{R}^n_{++}$ in at most $\log_{1.5}(1/\max(\delta(L \cap \mathbb{R}^n_{++}), \delta(L^\perp \cap \mathbb{R}^n_{++})))$ main iterations provided $L \cap \mathbb{R}^n_{++} \neq \emptyset$ or $L^\perp \cap \mathbb{R}^n_{++} \neq \emptyset$.

Algorithm 2 Extended Projection and Rescaling Algorithm

1 (**Initialization**)
Let $P_L \in \mathbb{R}^{n \times n}$ be the orthogonal projection onto L.
Let $P_{L^\perp} \in \mathbb{R}^{n \times n}$ be the orthogonal projection onto $L^\perp$.
Let $D := I$ and $P := P_L$.
Let $\hat{D} := I$ and $\hat{P} := P_{L^\perp}$.
2 (**Basic Procedure**)
Find $z \gneqq 0$ such that either $Pz > 0$ or $\|(Pz)^+\|_2 \leq \dfrac{1}{3\sqrt{n}}\|z\|_\infty$.
Find $\hat{z} \gneqq 0$ such that either $\hat{P}\hat{z} > 0$ or $\|(\hat{P}\hat{z})^+\|_2 \leq \dfrac{1}{3\sqrt{n}}\|\hat{z}\|_\infty$.
3 **if** $Pz > 0$ **then** HALT and **return** $x = D^{-1}Pz \in L \cap \mathbb{R}^n_{++}$ **end if**
4 **if** $\hat{P}\hat{z} > 0$ **then** HALT and **return** $\hat{x} = \hat{D}^{-1}\hat{P}\hat{z} \in L^\perp \cap \mathbb{R}^n_{++}$ **end if**
5 (**Rescaling step**)
Pick i such that $z_i = \|z\|_\infty$.
Put $D := (I + e_i e_i^{\mathrm{T}})D$ and $P := P_{DL}$.
Pick j such that $\hat{z}_j = \|\hat{z}\|_\infty$.
Put $\hat{D} := (I + e_j e_j^{\mathrm{T}})\hat{D}$ and $P := P_{\hat{D}L^\perp}$.
Go back to step 2.

We conclude this section by noting that the stopping condition $\|(Pz)^+\|_2 \leq \frac{1}{3\sqrt{n}}\|z\|_\infty$ in the basic procedure can be replaced by the less stringent condition $\|(Pz)^+\|_1 \leq \frac{1}{2}\|z\|_\infty$. This is closer to the approach used by Chubanov [8]. With this substitution it follows that if $L \cap \mathbb{R}^n_{++} \neq \emptyset$ then the algorithm finds $x \in L \cap \mathbb{R}^n_{++}$ in

at most $\log_2(1/\delta_\infty(L \cap \mathbb{R}^n_{++}))$ main iterations, where

$$\delta_\infty(L \cap \mathbb{R}^n_{++}) := \max_x \left\{ \prod_{i=1}^n x_i : x \in L \cap \mathbb{R}^n_{++}, \|x\|_\infty = 1 \right\}.$$

We chose to state the above Projection and Rescaling Algorithm with the stopping condition $\|(Pz)^+\|_2 \leq \frac{1}{3\sqrt{n}}\|z\|_\infty$ and presented the above statements in terms of $\delta(L \cap \mathbb{R}^n_{++})$ because this approach has a more natural extension to symmetric cones.

3 Extension to semidefinite conic systems

Let $\mathbb{S}^n$ denote the space of $n \times n$ real symmetric matrices. Assume $L \subseteq \mathbb{S}^n$ is a linear subspace and consider the problem

$$\text{find } X \in L \cap \mathbb{S}^n_{++}, \tag{7}$$

where $\mathbb{S}^n_{++}$ is the set of positive definite matrices, that is, the interior of the cone $\mathbb{S}^n_+$ of positive semidefinite matrices.

Assume the space $\mathbb{S}^n$ is endowed with the trace inner product

$$X \bullet S = \langle X, S \rangle := \text{trace}(XS).$$

Let $P_L : \mathbb{S}^n \to \mathbb{S}^n$ be the orthogonal projection onto L with respect to the trace inner product. Then (7) is equivalent to

$$\text{find } Y \in \mathbb{S}^n \text{ such that } P_L Y \in \mathbb{S}^n_{++}.$$

For $X \in \mathbb{S}^n$ let $\lambda(X) \in \mathbb{R}^n$ denote the vector of eigenvalues of X. We will rely on the Frobenius norm $\|X\|_F := \sqrt{\langle X, X \rangle} = \|\lambda(X)\|_2$ as well as on the operator norm $\|X\| := \max_{\|u\|_2=1} \|Xu\|_2 = \|\lambda(X)\|_\infty$.

Consider the following kind of *condition measure* of the set $L \cap \mathbb{S}^n_{++}$:

$$\delta(L \cap \mathbb{S}^n_{++}) := \max_X \{\det(X) : X \in L \cap \mathbb{S}^n_{++}, \|X\|_F^2 = n\}.$$

In analogy to the case discussed in the previous section, $L \cap \mathbb{S}^n_{++} \neq \emptyset$ implies that $\delta(L \cap \mathbb{S}^n_{++}) \in (0, 1]$ and $\delta(L \cap \mathbb{S}^n_{++}) = 1$ precisely when $I \in L \cap \mathbb{S}^n_{++}$.

For $X \in \mathbb{S}^n$ let X^+ denote the projection of X on $\mathbb{S}^n_+$. It is known, and easy to show, that if $X = Q\text{diag}(\lambda(X))Q^{\mathrm{T}}$ is the spectral decomposition of X then $X^+ = Q\text{diag}(\lambda(X)^+)Q^{\mathrm{T}}$.

The key property stated as Proposition 1 above extends as follows.

Proposition 2 *Assume $Z \gneq 0$ is such that $\|(P_L Z)^+\|_F \leq \epsilon \|Z\|$. Let $u \in \mathbb{R}^n, \|u\|_2 = 1$ be an eigenvector of Z with eigenvalue $\lambda_{\max}(Z) = \|Z\|$. Let $D : \mathbb{S}^n \to \mathbb{S}^n$ be defined as*

$$D(X) = (I + auu^{\mathrm{T}})X(I + auu^{\mathrm{T}})$$

for some constant $a > 0$. Then

$$\delta(D(L) \cap \mathbb{S}^n_{++}) \geq \frac{(1+a)^2}{(1+(2a+a^2)\epsilon)^n} \cdot \delta(L \cap \mathbb{S}^n_{++}).$$

In particular, if $\epsilon = \frac{1}{4n}$ and $a = \sqrt{2} - 1$ then

$$\delta(D(L) \cap \mathbb{S}^n_{++}) \geq 1.5 \cdot \delta(L \cap \mathbb{S}^n_{++}).$$

Proof: Observe that for $X \in L \cap \mathbb{S}^n_{++}$ the point $\hat{X} := \frac{\sqrt{n}}{\|D(X)\|_F} \cdot D(X)$ satisfies $\hat{X} \in D(L) \cap \mathbb{S}^n_{++}$ and $\|\hat{X}\|_F^2 = n$. Thus it suffices to show that for $X \in L \cap \mathbb{S}^n_{++}$ with $\|X\|_F^2 = n$ both $\det(D(X)) = (1+a)^2 \det(X)$ and $\|D(X)\|_F^2 \leq n(1+(2a+a^2)\epsilon)^2$ as this would imply $\det(\hat{X}) \geq \frac{(1+a)^2}{(1+(2a+a^2)\epsilon^2)^n} \cdot \det(X)$. Equivalently, it suffices to show that for $X \in L \cap \mathbb{S}^n_{++}$ with $\|X\|_F = 1$ both $\det(D(X)) = (1+a)^2 \det(X)$ and $\|D(X)\|_F^2 \leq (1+(2a+a^2)\epsilon)^2$.

Assume $X \in L \cap \mathbb{S}^n_{++}$ with $\|X\|_F = 1$ is fixed. Since $D(X) = (I + auu^{\mathrm{T}})X(I + auu^{\mathrm{T}})$ and $\|u\|_2 = 1$, it readily follows that

$$\det(D(X)) = \det(I + auu^{\mathrm{T}})^2 \det(X) = (1+a)^2 \det(X).$$

The first step above holds because $\det(AB) = \det(A)\det(B)$ for all $A, B \in \mathbb{S}^n$. The second step holds because $\det(I + auu^{\mathrm{T}}) = 1 + au^{\mathrm{T}}u = 1 + a$.

On the other hand,

$$\begin{aligned}
\|D(X)\|_F^2 &= \mathrm{trace}(D(X)^2) \\
&= \mathrm{trace}(X(I + auu^{\mathrm{T}})^2 X(I + auu^{\mathrm{T}})^2) \qquad (8)\\
&= \mathrm{trace}(X^2 + 2(2a+a^2)X^2uu^{\mathrm{T}} + (2a+a^2)^2(u^{\mathrm{T}}Xu)Xuu^{\mathrm{T}}) \\
&= \mathrm{trace}(X^2) + 2(2a+a^2)\mathrm{trace}(X^2uu^{\mathrm{T}}) + (2a+a^2)^2(u^{\mathrm{T}}Xu)\mathrm{trace}(Xuu^{\mathrm{T}}) \\
&= \mathrm{trace}(X^2) + 2(2a+a^2)\mathrm{trace}(u^{\mathrm{T}}X^2u) + (2a+a^2)^2(u^{\mathrm{T}}Xu)\mathrm{trace}(u^{\mathrm{T}}Xu) \\
&= \mathrm{trace}(X^2) + 2(2a+a^2)u^{\mathrm{T}}X^2u + (2a+a^2)^2(u^{\mathrm{T}}Xu)^2.
\end{aligned}$$

The steps above hold because $\mathrm{trace}(AB) = \mathrm{trace}(BA)$, $\mathrm{trace}(A+B) = \mathrm{trace}(A) + \mathrm{trace}(B)$, and $\mathrm{trace}(cA) = c \cdot \mathrm{trace}(A)$ for all $A, B \in \mathbb{S}^n$ and $c \in \mathbb{R}$.

Now observe that by construction $uu^{\mathrm{T}} \preceq \frac{Z}{\|Z\|}$. Thus using that $X \in L \cap \mathbb{S}^n_{++}$ and $\|X\|_F = 1$ we get

$$u^{\mathrm{T}}Xu = X \bullet uu^{\mathrm{T}} \leq \frac{X \bullet Z}{\|Z\|} = \frac{P_L X \bullet Z}{\|Z\|} = \frac{X \bullet P_L Z}{\|Z\|} \leq \frac{\|X\|_F \|(P_L Z)^+\|_F}{\|Z\|} \leq \epsilon.$$

Furthermore, since $X \in \mathbb{S}^n_{++}$ and $\|X\|_F = 1$, it follows that $X - X^2 \in \mathbb{S}^n_{++}$ and so $u^{\mathrm{T}}X^2u \leq u^{\mathrm{T}}Xu \leq \epsilon$. Hence (8) yields

$$\|D(X)\|_F^2 \leq 1 + 2(2a+a^2)\epsilon + (2a+a^2)^2\epsilon^2 = (1+(2a+a^2)\epsilon)^2.$$

■

The Rescaling and Projection Algorithm from Section 2, namely Algorithm 1, extends to Algorithm 3.

Algorithm 3 Projection and Rescaling Algorithm for Semidefinite Conic Systems

1 (**Initialization**)
Let $P_L : \mathbb{S}^n \to \mathbb{S}^n$ be the orthogonal projection onto L.
Let $D : \mathbb{S}^n \to \mathbb{S}^n$ be the identity map, $P := P_L$, and $a := \sqrt{2} - 1$.
2 (**Basic Procedure**)
Find $Z \gneqq 0$ such that either $P(Z) \succ 0$ or $\| (P(Z))^+ \|_F \leq \frac{1}{4n} \|Z\|$.
3 **if** $P(Z) \succ 0$ **then** HALT and **return** $X = D^{-1}(P(Z)) \in L \cap \mathbb{S}^n_{++}$ **end if**
4 (**Rescaling step**)
Pick $u \in \mathbb{R}^n, \|u\|_2 = 1$ such that $Zu = \lambda_{\max}(Z)u$.
Replace $D : \mathbb{S}^n \to \mathbb{S}^n$ with the mapping $X \mapsto (I + auu^{\mathrm{T}})D(X)(I + auu^{\mathrm{T}})$.
Let $P := P_{D(L)}$.
Go back to step 2.

Theorem 1 and its proof readily extends as follows.

Theorem 2 *If $L \cap \mathbb{S}^n_{++} \neq \emptyset$ then Algorithm 3 finds $X \in L \cap \mathbb{S}^n_{++}$ in at most $\log_{1.5}(1/\delta(L \cap \mathbb{S}^n_{++}))$ main iterations.*

As it was the case in Algorithm 1, the bulk of the work in each main iteration of Algorithm 3 is a call to the basic procedure. As we detail in Section 5, the simplest implementations of the basic procedure are guaranteed to terminate in $\mathcal{O}(n^4)$ perceptron or von Neumann steps. Each of these steps requires an operation of the form $Z \mapsto P(Z)$ in addition to a leading eigenvalue computation for a matrix in $\mathbb{S}^n$ and some other negligible computations. Assuming that P is maintained via an orthogonal basis for $D(L)$ each operation $Z \mapsto P(Z)$ requires $\mathcal{O}(mn^2)$ arithmetic operations where $m = \dim(L)$. The operation $Z \mapsto P(Z)$ dominates the leading eigenvalue computation. Indeed, there are several methods from the numerical linear algebra literature (see, e.g., [17]) that compute the leading eigenvalue and eigenvector of an $n \times n$ symmetric matrix in $\mathcal{O}(n^2)$ arithmetic operations. It thus follows that the total number of arithmetic operations required by Algorithm 3 is bounded above by

$$\mathcal{O}(mn^2 \cdot n^4 \cdot \log(1/\delta(L \cap \mathbb{S}^n_{++}))) = \mathcal{O}(mn^6 \log(1/\delta(L \cap \mathbb{S}^n_{++}))).$$

Algorithm 3 extends in the same fashion as Algorithm 1 extends to Algorithm 2 to find a point in either $L \cap \mathbb{S}^n_{++}$ or $L^\perp \cap \mathbb{S}^n_{++}$ provided one of them is feasible.

4 Extension to symmetric conic systems

Consider the problem

$$\text{find } x \in L \cap \Omega, \tag{9}$$

where $L \subseteq V$ and $\Omega \subseteq V$ are respectively a linear subspace and the interior of a symmetric cone in a finite-dimensional vector space V over $\mathbb{R}$.

We next present a version of the Projection and Rescaling Algorithm for the more general problem (9). To that end, we rely on some machinery of Euclidean Jordan Algebras. For succinctness we recall only the essential facts and pieces of notation that are necessary for our exposition. We refer the reader to the articles [23, 24] and the textbooks [3, 13] for a more detailed discussion of Euclidean Jordan algebras and their connection to optimization. The key connection between symmetric cones and Euclidean Jordan algebras is given by a theorem of Koecher and Vinberg that establishes a correspondence between symmetric cones and cones of squares of Euclidean Jordan algebras [13, Chapter III].

Assume V is endowed with a bilinear operation $\circ : V \times V \to V$ and $e \in V$ is a particular element of V. The triple $(V, \circ, e)$ is an *Euclidean Jordan algebra with identity element* if the following conditions hold:

- $x \circ y = y \circ x$ for all $x, y \in V$
- $x \circ (x^2 \circ y) = x^2 \circ (x \circ y)$ for all $x, y \in V$, where $x^2 = x \circ x$
- $x \circ e = x$ for all $x \in V$
- There exists an associative positive definite bilinear form on V.

Example 1 below summarizes the most popular types of Euclidean Jordan algebras used in optimization.

An element $c \in V$ is *idempotent* if $c^2 = c$. An idempotent element of V is a *primitive idempotent* if it is not the sum of two other idempotents. The rank r of V is the smallest integer such that for all $x \in V$ the set $\{e, x, x^2, \ldots, x^r\}$ is linearly dependent. Every element $x \in V$ has a *spectral decomposition*

$$x = \sum_{i=1}^{r} \lambda_i(x) c_i, \tag{10}$$

where $\lambda_i(x) \in \mathbb{R},\ i = 1, \ldots, r$ are the *eigenvalues* of x and $\{c_1, \ldots, c_r\}$ is a *Jordan frame,* that is, a collection of non-zero primitive idempotents such that $c_i \circ c_j = 0$ for $i \neq j$, and $c_1 + \cdots + c_r = e$. We will rely on the following simple observation: Given the spectral decomposition (10), we have $x \circ c_i = \lambda_i(x) c_i,\ i = 1, \ldots, r$.

The *trace* and *determinant* of $x \in V$ are respectively defined as $\text{trace}(x) = \sum_{i=1}^{r} \lambda_i(x)$ and $\det(x) = \prod_{i=1}^{r} \lambda_i(x)$. Throughout this section we assume that $(V, \circ, e)$

is an Euclidean Jordan algebra with identity. Furthermore, we assume that V is endowed with the following trace inner product:

$$\langle x, y \rangle := \operatorname{trace}(x \circ y). \tag{11}$$

We also assume that Ω is the interior of the cone of squares in V, that is, $\Omega = \operatorname{int}(\{x^2 : x \in V\})$.

Example 1 *The following are the most popular Euclidean Jordan algebras used in optimization.*

(a) The space $\mathbb{S}^n$ of $n \times n$ real symmetric matrices with the bilinear operation

$$X \circ Y := \frac{XY + YX}{2}$$

is an Euclidean Jordan algebra of rank n and identity element I. In this case, the spectral decomposition, trace, and determinant are precisely the usual ones. The cone of squares is the cone of positive semidefinite matrices $\mathbb{S}^n_+$.

(b) The space $\mathbb{R}^n$ with the bilinear operation

$$x \circ y = \begin{bmatrix} x_1 y_1 \\ \vdots \\ x_n y_n \end{bmatrix}$$

is an Euclidean Jordan algebra of rank n and identity element $e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$. In this case, the spectral decomposition of an element $x \in \mathbb{R}^n$ is

$$x = \sum_{i=1}^{n} x_i e_i.$$

For $x \in \mathbb{R}^n$ we have $\operatorname{trace}(x) = \sum_{i=1}^n x_i$ and $\det(x) = \prod_{i=1}^n x_i$. The cone of squares is the non-negative orthant $\mathbb{R}^n_+$.

(c) The space $\mathbb{R}^n$ with the bilinear operation

$$\begin{bmatrix} x_0 \\ \bar{x} \end{bmatrix} \circ \begin{bmatrix} y_0 \\ \bar{y} \end{bmatrix} := \begin{bmatrix} x^{\mathrm{T}} y \\ x_0 \bar{y} + y_0 \bar{x} \end{bmatrix}$$

is an Euclidean Jordan algebra of rank 2 and identity element $e = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}$. In this case, the spectral decomposition of an element $x = \begin{bmatrix} x_0 \\ \bar{x} \end{bmatrix} \in \mathbb{R}^n$ is

$$x = (x_0 + \|\bar{x}\|_2) \begin{bmatrix} 1/2 \\ \bar{u}/2 \end{bmatrix} + (x_0 - \|\bar{x}\|_2) \begin{bmatrix} 1/2 \\ -\bar{u}/2 \end{bmatrix},$$

where $\bar{u} \in \mathbb{R}^{n-1}$ *is such that* $\|\bar{u}\|_2 = 1$ *and* $\bar{x} = \|\bar{x}\|_2 \bar{u}$. *Consequently, for* $x \in V$ *we have* $\operatorname{trace}(x) = 2x_0$ *and* $\det(x) = x_0^2 - \|\bar{x}\|^2$. *The cone of squares is the second order cone* $\mathbb{L}_n = \left\{ x = \begin{bmatrix} x_0 \\ \bar{x} \end{bmatrix} \in \mathbb{R}^n : x_0 \geq \|\bar{x}\|_2 \right\}$.

(d) *A direct product of finitely many of the above types of Euclidean Jordan algebras is again an Euclidean Jordan algebra.*

Let $P_L : V \to V$ be the projection map onto L relative to the inner product defined in (11). Then (9) is equivalent to

$$\text{find } y \in V \text{ such that } P_L y \in \Omega.$$

We will rely on the Frobenius norm $\|x\|_F := \sqrt{\langle x, x \rangle} = \|\lambda(x)\|_2$, as well as on the operator norm $\|x\| := \|\lambda(x)\|_\infty$, where $\lambda(x) \in \mathbb{R}^r$ denotes the vector of eigenvalues of $x \in V$.

Consider the following kind of *condition measure* of the set $L \cap \Omega$:

$$\delta(L \cap \Omega) := \max_x \left\{ \det(x) : x \in L \cap \Omega, \|x\|_F^2 = r \right\}.$$

Observe that this condition measure matches the ones defined in Section 2 and Section 3 for the special cases $\Omega = \mathbb{R}^n_{++}$ and $\Omega = \mathbb{S}^n_{++}$. As in those special cases, observe that $L \cap \Omega \neq \emptyset$ implies $\delta(L \cap \Omega) \in (0, 1]$ with equality precisely when $e \in L \cap \Omega$.

Let $\bar{\Omega}$ denote the closure of Ω. For $v \in V$ let v^+ denote the projection of v on $\bar{\Omega}$. It is easy to see that if $v = \sum_{i=1}^r \lambda_i(v) c_i$ is the spectral decomposition of v, then $v^+ = \sum_{i=1}^r \lambda_i(v)^+ c_i$, where $\lambda_i(v)^+ = \max\{\lambda_i(v), 0\},\ i = 1, \ldots, r$.

Assume $c \in V$ is a primitive idempotent and $a > 0$ is a positive constant. The following mapping associated to c is key to our development. Let $D_v : V \to V$ be the *quadratic mapping* associated to $v = e + ac$, that is,

$$D_v x = 2v \circ (v \circ x) - v^2 \circ x. \tag{12}$$

The following identities readily follow from the properties of the Jordan algebra product

$$v \circ x = (e + ac) \circ x = x + ac \circ x$$

$$v \circ (v \circ x) = (e + ac) \circ (x + ac \circ x) = x + 2ac \circ x + a^2 c \circ (c \circ x)$$

$$v^2 \circ x = (e + 2ac + a^2 c) \circ x = x + (2a + a^2) c \circ x.$$

Hence the quadratic mapping associated to $v = e + ac$ defined in (12) can also be written as

$$D_v x = x + (2a - a^2) c \circ x + 2a^2 c \circ (c \circ x). \tag{13}$$

Proposition 3 *Assume $z \in \bar{\Omega} \setminus \{0\}$ is such that $\| (P_L z)^+ \|_F \leq \epsilon \|z\|$. Let $c \in V$ be a primitive idempotent such that $z \circ c = \lambda_{\max}(z)c$ and let $D_v : V \to V$ be the quadratic mapping associated to $v = e + ac$ as in (12) for some constant $a > 0$. Then*

$$\delta(D_v(L) \cap \Omega) \geq \frac{(1+a)^2}{(1+(2a+a^2)\epsilon)^r} \cdot \delta(L \cap \Omega).$$

In particular, if $\epsilon = \frac{1}{4r}$ and $a = \sqrt{2} - 1$ then

$$\delta(D_v(L) \cap \Omega) \geq 1.5 \cdot \delta(L \cap \Omega).$$

Proof: Observe that for $x \in L \cap \Omega$ the point $\hat{x} := \frac{\sqrt{r}}{\|D_v x\|_F} \cdot D_v x$ satisfies $\hat{x} \in D_v(L) \cap \Omega$ and $\|\hat{x}\|_F^2 = r$. Thus it suffices to show that for $x \in L \cap \Omega$ with $\|x\|_F^2 = r$ both $\det(D_v x) = (1+a)^2 \det(x)$ and $\|D_v x\|_F^2 \leq r(1+(2a+a^2)\epsilon)^2$ as this would imply $\det(\hat{x}) \geq \frac{(1+a)^2}{(1+(2a+a^2)\epsilon)^r} \det(x)$. Equivalently, it suffices to show that for $x \in L \cap \Omega$ with $\|x\|_F = 1$ both $\det(D_v x) \geq (1+a)^2 \det(x)$ and $\|D_v x\|_F \leq 1 + (2a+a^2)\epsilon$. Assume $x \in L \cap \Omega$ with $\|x\|_F = 1$ is fixed. Since D_v is the quadratic form associated to $v = e + ac$, it follows from [13, Prop III.4.2] or from [3, Prop 2.5.4] that

$$\det(D_v x) = \det(v)^2 \det(x) = (1+a)^2 \det(x).$$

On the other hand, the expression (13) for $D_v x$ yields

$$\|D_v x\|_F^2 = \|x\|_F^2 + 2(2a+a^2)\text{trace}(c \circ x^2) + (2a+a^2)^2 \text{trace}((c \circ x)^2). \tag{14}$$

Next observe that $\frac{z}{\|z\|} - c \in \bar{\Omega}$. Thus using that $x \in L \cap \Omega$ and $\|x\|_F = 1$ we get

$$\langle x, c \rangle \leq \frac{1}{\|z\|} \langle x, z \rangle = \frac{1}{\|z\|} \langle P_L x, z \rangle = \frac{1}{\|z\|} \langle x, P_L z \rangle \leq \frac{1}{\|z\|} \|x\|_F \|(P_L z)^+\|_F \leq \epsilon.$$

Since $c, x \in \Omega$ we have $c \circ x \in \Omega$. In particular, $\text{trace}((c \circ x)^2) \leq (\text{trace}(c \circ x))^2 \leq \epsilon^2$. Furthermore, since $x \in \Omega$ and $\|x\|_F = 1$ we also have $x - x^2 \in \Omega$. In particular $\text{trace}(c \circ x^2) \leq \text{trace}(c \circ x) \leq \epsilon$. Therefore (14) yields

$$\|D_v x\|_F \leq 1 + (2a+a^2)\epsilon.$$

■

We have the more generic version of the Projection and Rescaling Algorithm presented in Algorithm 4. Theorem 3 states the main property of Algorithm 4.

Theorem 3 *If $L \cap \Omega \neq \emptyset$ then Algorithm 4 finds $x \in L \cap \Omega$ in at most $\log_{1.5}(1/\delta(L \cap \Omega))$ main iterations.*

Algorithm 4 Projection and Rescaling Algorithm for Symmetric Conic Systems

1 (**Initialization**)
Let $P_L : V \to V$ be the orthogonal projection onto L.
Let $D : V \to V$ be the identity map, $P := P_L$, and $a := \sqrt{2} - 1$.
2 (**Basic Procedure**)
Find $z \in \bar{\Omega} \setminus \{0\}$ such that either $Pz \in \Omega$ or $\|(Pz)^+\|_F \leq \frac{1}{4r}\|z\|$.
3 **if** $Pz \in \Omega$ **then** HALT and **return** $x = D^{-1}Pz \in L \cap \Omega$ **end if**
4 (**Rescaling step**)
Pick $c \in V$ a primitive idempotent point such that $z \circ c = \lambda_{\max}(z)c$.
Let $D_v : V \to V$ be the quadratic mapping associated to $v = e + ac$.
Replace D with $D_v D$ and P with P_{DL}.
Go back to step 2.

We note that in the special case when $V = \mathbb{S}^n$ with the Euclidean Jordan algebra described in Example 1(a), Algorithm 4 reduces to Algorithm 3 in Section 3. On the other hand, when $V = \mathbb{R}^n$ with the Euclidean Jordan algebra described in Example 1(b), Algorithm 4 yields a slightly weaker version of Algorithm 1 in Section 2. It is the small price we pay for extending the algorithm to general symmetric cones.

Once again, the bulk of each main iteration in Algorithm 4 is a call to the basic procedure. As we detail in Section 5, for $r =$ Jordan algebra rank of V the simplest implementations of the basic procedure terminate in $\mathcal{O}(r^4)$ perceptron or von Neumann steps. Each of these steps requires an operation of the form $z \mapsto P(z)$ in addition to a Jordan leading eigenvalue computation in V and some negligible computations. The amount of computational work required by the operation $z \mapsto P(z)$ dominates that of the other operations. Assuming that P is maintained via an orthogonal basis for $D(L)$, it follows that the total number of arithmetic operations required by Algorithm 4 is bounded above by

$$\mathcal{O}(mn \cdot r^4 \cdot \log(1/\delta(L \cap \Omega)))$$

where $m = \dim(L)$, $n = \dim(V)$, and $r =$ Jordan algebra rank of V.

Algorithm 4 also extends in the same fashion as Algorithm 1 extends to Algorithm 2 to find a point in either $L \cap \Omega$ or $L^\perp \cap \Omega$ provided one of them is feasible.

5 The basic procedure

We next describe various possible implementations for the basic procedure, i.e., step 2 in Algorithm 4. The schemes we discuss below vary in their work per iteration and overall speed of convergence. Assume Ω and V are as in Section 4 and define the *spectraplex* $\Delta(\Omega)$ as follows:

$$\Delta(\Omega) := \left\{x \in \bar{\Omega} : \langle e, x\rangle = 1\right\}.$$

Assume also that $P : V \to V$ is a projection mapping.

5.1 Perceptron scheme

This is perhaps the simplest possible scheme. It is based on the classical perceptron algorithm of Rosenblatt [22], which has a natural extension to conic systems [6, 20]. We assume that the following kind of separation oracle for Ω is an available: Given $v \in V$, the separation oracle either verifies that $v \in \Omega$ or else it yields a separating vector $u \in \Delta(\Omega)$ such that $\langle u, v \rangle \leq 0$.

Observe that such a separation oracle is readily available when Ω is $\mathbb{R}^n_{++}, \mathbb{S}^n_{++}$, $\mathrm{int}(\mathbb{L}_n)$ or any direct product of these kinds of cones. Algorithm 5 gives an implementation of the basic procedure via the perceptron scheme.

Algorithm 5 Perceptron Scheme

Pick $z_0 \in \Delta(\Omega)$ and $t := 0$.
while $Pz_t \notin \Omega$ and $\|(Pz_t)^+\|_F > \frac{1}{4r}\|z_t\|$ **do**
  Pick $u \in \Delta(\Omega)$ such that $\langle u, Pz_t \rangle \leq 0$.
  Let $z_{t+1} := \left(1 - \frac{1}{t+1}\right) z_t + \frac{1}{t+1} u = \frac{t}{t+1} z_t + \frac{1}{t+1} u$.
  $t := t + 1$.
end while

Proposition 4 *If Algorithm 5 has not halted after $t \geq 1$ iterations then*

$$\|Pz_t\|_F^2 \leq \frac{1}{t}.$$

Proof: Proceed by induction on t. To that end, observe that $\|z\|_F \leq 1$ for all $z \in \Delta(\Omega)$ and so $\|Pz\|_F \leq 1$ since P is a projection. Therefore the condition readily holds for $t = 1$. Assume the condition holds for t and the algorithm continues to iteration $t + 1$. Then

$$\begin{aligned}
\|Pz_{t+1}\|_F^2 &= \frac{t^2}{(t+1)^2}\|Pz_t\|_F^2 + \frac{1}{(t+1)^2}\|Pu\|_F^2 + \frac{2t}{(t+1)^2}\langle u, Pz_t \rangle \\
&\leq \frac{t^2}{(t+1)^2}\|Pz_t\|_F^2 + \frac{1}{(t+1)^2}\|Pu\|_F^2 \\
&\leq \frac{t^2}{(t+1)^2}\frac{1}{t} + \frac{1}{(t+1)^2} \\
&= \frac{1}{t+1}.
\end{aligned}$$

■

Corollary 1 *If the basic procedure is implemented via Algorithm 5, then one of the stopping conditions $Pz \in \Omega$ or $\|(Pz)^+\|_F \leq \frac{1}{4r}\|z\|$ is reached after at most*

$(4r^2)^2 = 16r^4$ *iterations. In the special case* $\Omega = \mathbb{R}^n_{++}$ *one of the stopping conditions* $Pz > 0$ *or* $\|(Pz)^+\|_2 \leq \frac{1}{3\sqrt{n}}\|z\|_\infty$ *is reached after at most* $(3n\sqrt{n})^2 = 9n^3$ *iterations.*

Proof: Both statements readily follow from Proposition 4 and the observations that $\|z\| \geq \frac{1}{r}$ for all $z \in \Delta(\Omega)$ and $\|v^+\|_F \leq \|v\|_F$ for all $v \in V$. ■

5.2 Von Neumann scheme

The second scheme is based on a classical algorithm communicated by von Neumann to Dantzig and later studied by Dantzig in an unpublished manuscript [9].

Several authors have studied various aspects of this algorithm over the last few years [11, 18, 26]. The von Neumann scheme can be seen as a greedy variation of the perceptron scheme that includes an exact line-search in each iteration. In the special case $\Omega = \mathbb{R}^n_{++}$, this scheme is essentially the same as the basic procedure proposed by Chubanov [8].

Assume the following mapping $u : V \to \Delta(\Omega)$ is available:

$$u(v) := \operatorname*{argmin}_{u \in \Delta(\Omega)} \langle u, v \rangle .$$

Observe that such an mapping is readily available when Ω is $\mathbb{R}^n_{++}, \mathbb{S}^n_{++}$, $\mathrm{int}(\mathbb{L}_n)$ or any direct product of these kinds of cones. Algorithm 6 gives an implementation of the basic procedure via the von Neumann scheme.

Algorithm 6 Von Neumann Scheme

1 Pick $z_0 \in \Delta(\Omega)$ and $t := 0$.
2 **while** $Pz_t \notin \Omega$ and $\|(Pz_t)^+\|_F > \frac{1}{4r}\|z_t\|$ **do**
Let $u = u(Pz_t)$.
Let $z_{t+1} := z_t + \theta_t(u - z_t)$ where

$$\theta_t = \operatorname*{argmin}_{\theta \in [0,1]} \|P(z_t + \theta(u - z_t))\|_F^2 = \frac{\|Pz_t\|_F^2 - \langle u, Pz_t \rangle}{\|Pz_t\|_F^2 + \|Pu\|_F^2 - 2\langle u, Pz_t \rangle}.$$

$t := t + 1$.
3 **end while**

An inductive argument like the one used in the proof of Proposition 4 yields the following result. However, we note that the choice of u and θ_t at each iteration suggests that $\|Pz_t\|_F^2$ may decrease faster for this scheme than for the previous one.

Proposition 5 *If Algorithm 6 has not halted after* $t \geq 1$ *iterations then*

$$\|Pz_t\|_F^2 \leq \frac{1}{t}.$$

Corollary 2 *If the basic procedure is implemented via Algorithm 6, then one of the stopping conditions* $Pz \in \Omega$ *or* $\|(Pz)^+\|_F \leq \frac{1}{4r}\|z\|$ *is reached after at most* $(4r^2)^2 = 16r^4$ *iterations. In the special case* $\Omega = \mathbb{R}^n_{++}$ *one of the stopping conditions* $Pz > 0$ *or* $\|(Pz)^+\|_2 \leq \frac{1}{3\sqrt{n}}\|z\|_\infty$ *is reached after at most* $(3n\sqrt{n})^2 = 9n^3$ *iterations.*

Proof: Both statements readily follow from Proposition 5 and the observations that $\|z\| \geq \frac{1}{r}$ for all $z \in \Delta(\Omega)$ and $\|v^+\|_F \leq \|v\|_F$ for all $v \in V$. ■

In the special case $\Omega = \mathbb{R}^n_{++}$ Corollary 2 recovers the iteration bound $\mathcal{O}(n^3)$ originally given by Chubanov [8, Lemma 2.2].

5.3 Smooth perceptron scheme

Soheili and Peña [25, 26] proposed a variation of the perceptron that relies on the following tweaked version of the subproblem $\min_{u \in \Delta(\Omega)} \langle u, v \rangle$ used in the von Neumann scheme. Given $\mu > 0$ let $u_\mu : V \to \Delta(\Omega)$ be defined as

$$u_\mu(v) := \operatorname*{argmin}_{u \in \Delta(\Omega)} \left\{ \langle u, v \rangle + \frac{\mu}{2}\|u - \bar{u}\|^2 \right\}.$$

where $\bar{u} \in \Delta(\Omega)$ is a given point, e.g., $\bar{u} = \frac{1}{r}e$.

Peña and Soheili [26] show that the mapping u_μ is readily available when Ω is $\mathbb{R}^n_{++}, \mathbb{S}^n_{++}, \mathbb{L}_n$ or any direct product of these kinds of cones. Algorithm 7 gives an implementation of the basic procedure via the smooth perceptron scheme.

Algorithm 7 Smooth Perceptron Scheme

Let $u_0 := \bar{u}$; $\mu_0 = 2$; $z_0 := u_{\mu_0}(Pu_0)$; and $t := 0$.
while $Pu_t \notin \Omega$ and $\|(Pz_t)^+\|_F > \frac{1}{4r}\|z_t\|$ **do**
  $\theta_t := \frac{2}{t+3}$
  $u_{t+1} := (1 - \theta_t)(u_t + \theta_t z_t) + \theta_t^2 u_{\mu_t}(Pz_t)$
  $\mu_{t+1} = (1 - \theta_t)\mu_t$
  $z_{t+1} := (1 - \theta_t)z_t + \theta_t u_{\mu_{t+1}}(Pu_{t+1})$
  $t := t + 1$.
end while

Proposition 6 *If Algorithm 7 has not halted after* $t \geq 1$ *iterations then*

$$\|Pz_t\|_F^2 \leq \frac{8}{(t+1)^2}.$$

Proposition 6 follows from [26, Lemma 1]. For the sake of exposition, Lemma 1 below restates this technical result in the current context. To that end, define $\varphi : V \to \mathbb{R}$ as follows

$$\varphi(z) := -\frac{1}{2}\|Pz\|_F^2 + \min_{u \in \Delta(\Omega)} \langle u, Pz \rangle.$$

Observe that $Pz \in \Omega$ if $\varphi(z) > 0$. For $\mu > 0$ define $\varphi_\mu : V \to \mathbb{R}$ as follows

$$\varphi_\mu(z) := -\frac{1}{2}\|Pz\|_F^2 + \min_{u \in \Delta(\Omega)} \left\{ \langle u, Pz \rangle + \frac{\mu}{2}\|u - \bar{u}\|^2 \right\}.$$

Lemma 1 (from [26]) (a) *For all $\mu > 0$ and $z \in V$*

$$0 \leq \varphi_\mu(z) - \varphi(z) \leq \mu.$$

(b) *The iterates generated by Algorithm 7 satisfy*

$$\frac{1}{2}\|Pz_t\|_F^2 \leq \varphi_{\mu_t}(u_t).$$

Proof of Proposition 6. Since the algorithm has not halted after t iterations we have $Pu_t \notin \Omega$ and consequently $\varphi(u_t) \leq 0$. Thus Lemma 1 yields

$$\|Pz_t\|_F^2 \leq 2\varphi_{\mu_t}(u_t) \leq 2(\mu_t + \varphi(u_t)) \leq 2\mu_t.$$

To conclude, observe that $\mu_t = \frac{4}{(t+1)(t+2)} \leq \frac{4}{(t+1)^2}$. ■

Corollary 3 *If the basic procedure is implemented via Algorithm 7, then one of the stopping conditions $Pu \in \Omega$ or $\|Pz\|_F \leq \frac{1}{4r}\|z\|$ is reached after at most $8\sqrt{2}r^2 - 1$ iterations. In the special case $\Omega = \mathbb{R}^n_{++}$ one of the stopping conditions $Pz > 0$ or $\|(Pz)^+\|_2 \leq \frac{1}{3\sqrt{n}}\|z\|_\infty$ is reached after at most $6n\sqrt{2n} - 1$ iterations.*

Proof: Both statements follow from Proposition 6 and the observations that $\|z\| \geq \frac{1}{r}$ for all $z \in \Delta(\Omega)$ and $\|v^+\|_F \leq \|v\|_F$ for all $v \in V$. ■

The iteration bound $\mathcal{O}(r^2)$ for the smooth perceptron scheme versus the iteration bound $\mathcal{O}(r^4)$ for the perceptron scheme or von Neumann scheme does not account for the potentially higher cost of a smooth perceptron iteration. Hence we next provide a bound on the number of arithmetic operations. Aside from comparable operations of the form $z \mapsto P(z)$, each iteration of the smooth perceptron requires the computation of $u_\mu(v)$ versus the computation of $u(v)$ required by the perceptron or von Neumann schemes. As it is discussed in detail in [26], the computation of $u_\mu(v)$ requires a complete eigenvalue decomposition of $v + \mu\bar{u}$ whereas the computation of $u(v)$ only requires computing the smallest eigenvalue and corresponding eigenvector of $v + \mu\bar{u}$. By considering the special case of symmetric matrices, it follows that a complete Jordan eigenvalue decomposition requires $\mathcal{O}(r^3)$ arithmetic operations. Hence the number of arithmetic operations required by the smooth perceptron scheme is bounded above by

$$\mathcal{O}(\max(mn, r^3) \cdot r^2).$$

On the other hand, a smallest eigenvalue calculation requires $\mathcal{O}(r^2)$ arithmetic operations. Hence the number of arithmetic operations required by either the perceptron scheme or the von Neumann scheme is bounded above by

$$\mathcal{O}(\max(mn, r^2) \cdot r^4).$$

5.4 Von Neumann with away steps scheme

We now consider another variant on the von Neumann scheme that includes so-called *away steps*. This can be seen as a particular case of the Frank-Wolfe algorithm with away steps that has recently become a subject of renewed attention [2, 5, 14, 18]. The away steps rely on the following construction. Given $z \in \Delta(\Omega)$, let $z = \sum_{i=1}^{r} \lambda_i(z) c_i$ be the spectral decomposition of z and define the *support* of z as $S(z) := \{c_i : \lambda_i(z) > 0\}$. In principle we could update z by decreasing the weight on an element of $S(z)$ while increasing the other weights. Let

$$c(z) := \operatorname*{argmax}_{c \in S(z)} \langle c, Pz \rangle$$

and let $\lambda(z)$ denote the eigenvalue of $c(z)$ in the spectral decomposition of z. Algorithm 8 gives the implementation of the basic procedure via the von Neumann with away steps scheme.

Algorithm 8 Von Neumann with Away Steps Scheme

Pick $z_0 \in \Delta(\Omega)$ and $t := 0$.
while $Pz_t \notin \Omega$ and $\|(Pz_t)^+\|_F > \frac{1}{4r}\|z_t\|$ **do**
Let $u = u(Pz_t)$ and $c = c(z_t)$.
if $\|Pz_t\|^2 - \langle u, Pz_t \rangle > \langle c, Pz_t \rangle - \|Pz_t\|^2$ **then** (regular step)
$a := u - z_t$; $\theta_{\max} = 1$
else (away step)
$a := z_t - c$; $\theta_{\max} = \frac{\lambda(z)}{1-\lambda(z)}$
endif
Let $z_{t+1} := z_t + \theta_t a$ where

$$\theta_t = \operatorname*{argmin}_{\theta \in [0, \theta_{\max}]} \|P(z_t + \theta a)\|_F^2 = \min\left\{\theta_{\max}, -\frac{\langle z_t, Pa \rangle}{\|Pa\|_F^2}\right\}.$$

$t := t + 1$.
end while

Proposition 7 *If Algorithm 8 has not halted after $t \geq 1$ iterations then*

$$\|Pz_t\|_F^2 \leq \frac{8}{t}.$$

Proof: This readily follows via the same argument used in the proof of [18, Theorem 1(b)]. ■

Corollary 4 *If the basic procedure is implemented via Algorithm 8, then one of the stopping conditions $Pz \in \Omega$ or $\|(Pz)^+\|_F \leq \frac{1}{4r}\|z\|$ is reached after at most $8(4r^2)^2 = 128r^4$ iterations. In the special case $\Omega = \mathbb{R}^n_{++}$ one of the stopping conditions $Pz > 0$ or $\|(Pz)^+\|_2 \leq \frac{1}{3\sqrt{n}}\|z\|_\infty$ is reached after at most $8(3n\sqrt{n})^2 = 72n^3$ iterations.*

Proof: Both statements readily follow from Proposition 7 and the observations that $\|z\| \geq \frac{1}{r}$ for all $z \in \Delta(\Omega)$ and $\|v^+\|_F \leq \|v\|_F$ for all $v \in V$. ■

We note that although the bound in Proposition 7 is weaker than that in Proposition 4 and Proposition 5, the von Neumann with away steps scheme tends to generate iterates $z \in \Delta(\Omega)$ with smaller support. Since these kinds of points in $\Delta(\Omega)$ in turn tend to have a larger value of $\|z\|$, this could be an advantage as the scheme may reach the stopping condition $\|(Pz)^+\| \leq \frac{1}{4r}\|z\|$ sooner.

6 Updating the projection matrix

Each rescaling step requires the update of the projection matrix from P_L to $P_{D(L)}$. We next describe how this update can be performed. As the subsections below detail, in certain important cases this update can be done much more efficiently than simply performing a naive recalculation of the projection matrix.

A possible approach to maintaining and updating the projection matrix is via orthogonal bases. In particular, assume $P_L = QQ^{\mathrm{T}}$ for some matrix $Q \in \mathbb{R}^{n\times m}$ whose columns form an orthogonal basis of L, that is, $\mathrm{span}(Q) = L$ and $Q^{\mathrm{T}}Q = I_m$. To obtain a likewise expression $P_{D(L)} = \tilde{Q}\tilde{Q}^{\mathrm{T}}$ where the columns of $\tilde{Q}$ are an orthogonal basis of $D(L)$ we can proceed as follows.

First, observe that $\mathrm{span}(DQ) = D(L)$. Henceforth, it suffices to orthogonalize the columns of DQ. That is, we need to find $R \in \mathbb{R}^{m\times m}$ such that DQR is orthogonal, or equivalently such that

$$(DQR)^{\mathrm{T}}(DQR) = R^{\mathrm{T}}Q^{\mathrm{T}}D^{\mathrm{T}}DQR = I_m. \tag{15}$$

Although a matrix R such that (15) holds could be achieved via a Gram-Schmidt procedure for the columns of DQ or via a Cholesky factorization of $Q^{\mathrm{T}}D^{\mathrm{T}}DQ$, the particular structure of D may enable a more efficient procedure. In all of the cases discussed above D is of the form $I_n + B$ for some structured and symmetric matrix $B \in \mathbb{R}^{n\times n}$. In this case

$$Q^{\mathrm{T}}D^{\mathrm{T}}DQ = Q^{\mathrm{T}}(I_n + 2B + B^2)Q = I_m + Q^{\mathrm{T}}(2B + B^2)Q.$$

Let $Q^{\mathrm{T}}(2B + B^2)Q = P\Lambda P^{\mathrm{T}}$ be the spectral decomposition of $Q^{\mathrm{T}}(2B + B^2)Q$ for some orthogonal matrix $P \in \mathbb{R}^{m\times p}$ and some diagonal matrix $\Lambda \in \mathbb{R}^{p\times p}$. It readily follows that (15) holds for

$$R = I_m - P\bar{\Lambda}P^{\mathrm{T}}$$

if $\bar{\Lambda} \in \mathbb{R}^{p\times p}$ is a diagonal matrix such that

$$(I_m - P\bar{\Lambda}P^{\mathrm{T}})(I_m + P\Lambda P^{\mathrm{T}})(I_m - P\bar{\Lambda}P^{\mathrm{T}}) = I_m. \tag{16}$$

Observe that (16) holds provided the diagonal matrix $\bar{\Lambda}$ solves

$$-2\bar{\Lambda} + \Lambda - 2\bar{\Lambda}\Lambda + \bar{\Lambda}^2 + \bar{\Lambda}^2\Lambda = 0.$$

One of the solutions of this equation is

$$\bar{\Lambda} = (I_p + \Lambda)^{-1/2} + I_p. \tag{17}$$

Notice that $\bar{\Lambda}$ is easily computable componentwise since Λ is a diagonal matrix.

6.1 The case $\Omega = \mathbb{R}^n_{++}$

In this case D is of the form $D = I + e_i e_i^{\mathrm{T}}$. In this case $B = e_i e_i^{\mathrm{T}}$ and the term $Q^{\mathrm{T}}(2B + B^2)Q$ turns out to be

$$3Q^{\mathrm{T}} e_i e_i^{\mathrm{T}} Q = 3q_i q_i^{\mathrm{T}}$$

where $q_i = Q^{\mathrm{T}} e_i \in \mathbb{R}^m$. The spectral decomposition of $Q^{\mathrm{T}}(2B + B^2)Q = 3q_i q_i^{\mathrm{T}}$ is

$$\frac{q_i}{\|q_i\|} \cdot (3\|q_i\|^2) \cdot \frac{q_i^{\mathrm{T}}}{\|q_i\|}.$$

Hence

$$R = I_m - \frac{q_i}{\|q_i\|} \cdot \left(1 + \frac{1}{\sqrt{1 + 3\|q_i\|^2}}\right) \cdot \frac{q_i^{\mathrm{T}}}{\|q_i\|} = I_m - \left(1 + \frac{1}{\sqrt{1 + 3\|q_i\|^2}}\right) \cdot \frac{q_i q_i^{\mathrm{T}}}{\|q_i\|^2}$$

and so

$$\tilde{Q} = (I_n + e_i e_i^{\mathrm{T}})Q \left(I_m - \left(1 + \frac{1}{\sqrt{1 + 3\|q_i\|^2}}\right) \cdot \frac{q_i q_i^{\mathrm{T}}}{\|q_i\|^2}\right).$$

6.2 The case $\Omega = \mathbb{S}^n_{++}$

Assume the "columns" of Q correspond to the matrices $A_1, \ldots, A_m \in \mathbb{S}^n$ such that $A_i \bullet A_i = 1,\ i = 1, \ldots, m$ and $A_i \bullet A_j = 0, i \neq j$. The columns of the new DQ correspond to the matrices

$$D(A_i) = (I_n + auu^{\mathrm{T}})A_i(I_n + auu^{\mathrm{T}}) = A_i + auu^{\mathrm{T}}A_i + aA_iuu^{\mathrm{T}} + a^2(u^{\mathrm{T}}A_iu)uu^{\mathrm{T}}.$$

In particular, the columns of the new $BQ = DQ - Q$ are

$$B(A_i) = auu^{\mathrm{T}}A_i + aA_iuu^{\mathrm{T}} + a^2(u^{\mathrm{T}}A_iu)uu^{\mathrm{T}}.$$

Next, observe that

$$B(A_i) \bullet A_j = 2a(A_iu)^{\mathrm{T}}(A_ju) + a^2(u^{\mathrm{T}}A_iu)(u^{\mathrm{T}}A_ju)$$

and

$$B(A_i) \bullet B(A_j) = 2a^2(A_iu)^{\mathrm{T}}(A_ju) + (2a^2 + 4a^3 + a^4)(u^{\mathrm{T}}A_iu)(u^{\mathrm{T}}A_ju).$$

Consequently, the (i,j) entry of the matrix $Q(2B+B^2)Q^{\mathrm{T}}$ is

$$2(2a+a^2)(A_iu)^{\mathrm{T}}(A_ju)+(2a+a^2)^2(u^{\mathrm{T}}A_iu)(u^{\mathrm{T}}A_ju).$$

Therefore

$$Q^{\mathrm{T}}(2B+B^2)Q=UWU^{\mathrm{T}},$$

where

$$U^{\mathrm{T}}=\begin{bmatrix} A_1u & A_2u & \cdots & A_mu \\ u^{\mathrm{T}}A_1u & u^{\mathrm{T}}A_2u & \cdots & u^{\mathrm{T}}A_mu \end{bmatrix},\ W=\begin{bmatrix} 2(2a+a^2) & 0 \\ 0 & (2a+a^2)^2 \end{bmatrix}. \tag{18}$$

When $m\leq n$, it is typically cheaper to compute $R=L^{-\mathrm{T}}$ via the Cholesky factorization $LL^{\mathrm{T}}=I_m+UWU^{\mathrm{T}}$ of $Q^{\mathrm{T}}D^{\mathrm{T}}DQ=I_m+UWU^{\mathrm{T}}$. On the other hand, if $m\gg n+1$, it is typically more efficient to find the spectral decomposition $P\Lambda P^{\mathrm{T}}=UWU^{\mathrm{T}}$ for some orthogonal matrix $P\in\mathbb{R}^{m\times p}$ and some diagonal matrix $\Lambda\in\mathbb{R}^{p\times p}$, and then compute $R=I_m-P\bar{\Lambda}P^{\mathrm{T}}$ where $\bar{\Lambda}=(I_p+\Lambda)^{-1/2}+I_p$. In either case, it follows that the columns of DQR form an orthogonal basis for $D(L)$.

6.3 The case $\Omega=\mathbf{int}(\mathbb{L}_n)$

In this case D is the matrix representation of the mapping

$$x\mapsto x+(2a-a^2)c\circ x+2a^2c\circ(c\circ x)$$

where $c=\frac{1}{2}\begin{bmatrix}1\\ \bar{u}\end{bmatrix}$, with $\bar{u}\in\mathbb{R}^{n-1}$, $\|\bar{u}\|_2=1$. Observe that the mapping $x\mapsto c\circ x$ can be written as

$$x\mapsto\frac{1}{2}\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & I\end{bmatrix}x.$$

It thus follows that $D=I+B$ where

$$B=\frac{2a-a^2}{2}\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & I\end{bmatrix}+\frac{2a^2}{4}\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & I\end{bmatrix}^2=a\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & I\end{bmatrix}+\frac{a^2}{2}\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & \bar{u}\bar{u}^{\mathrm{T}}\end{bmatrix}.$$

Therefore,

$$B^2=a^2\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & I\end{bmatrix}+\left(a^2+2a^3+\frac{a^4}{2}\right)\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & \bar{u}\bar{u}^{\mathrm{T}}\end{bmatrix}$$

and

$$2B+B^2=(2a+a^2)\left\{\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & I\end{bmatrix}+\frac{2a+a^2}{2}\begin{bmatrix}1 & \bar{u}^{\mathrm{T}}\\ \bar{u} & \bar{u}\bar{u}^{\mathrm{T}}\end{bmatrix}\right\}. \tag{19}$$

In particular, $Q^{\mathrm{T}}(2B+B^2)Q$ is easily computable. This computation provides the basis for the more interesting case when Ω is a direct product of semidefinite and second-order cones that we discussed next.

6.4 Direct products of semidefinite and second-order cones

We now consider the case $\Omega = K_1 \times \cdots \times K_r \subseteq \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_r}$ where each $K_i \subseteq \mathbb{R}^{n_i}$ is a semidefinite cone or a second-order cone. Assume $\mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_r}$ is endowed with the appropriate Euclidean Jordan algebra structure. It is easy to see that a primitive idempotent in this vector space is of the form $\begin{bmatrix} 0 & \cdots & c_i^{\mathrm{T}} & \cdots & 0 \end{bmatrix}^{\mathrm{T}}$ where v_i is a primitive idempotent in $\mathbb{R}^{n_i}$. It follows that the scaling matrix D is of the form

$$D = \begin{bmatrix} D_1 & & \\ & \ddots & \\ & & D_r \end{bmatrix}$$

where $D_j = I_{n_j}$ for $j \neq i$ and $D_i = I_{n_i} + B$ for some structured and symmetric matrix $B \in \mathbb{R}^{n_i \times n_i}$ that depends on the idempotent c_i. Observe that $Q^{\mathrm{T}} = \begin{bmatrix} Q_1^{\mathrm{T}} & \cdots & Q_r^{\mathrm{T}} \end{bmatrix}$ where each $Q_j \in \mathbb{R}^{n_j \times m}$. It thus follows that

$$Q^{\mathrm{T}} D^{\mathrm{T}} D Q = I_m + Q_i^{\mathrm{T}}(2B + B^2)Q_i.$$

The particular expression for the term $Q_i^{\mathrm{T}}(2B + B^2)Q_i$ is of the form (18) in Section 6.2 or of the form $Q_i^{\mathrm{T}}(2B + B^2)Q_i$ where $2B + B^2$ is as in (19) in Section 6.3.

Again as we mentioned in Section 6.2 and in Section 6.3, when $m \leq n_i$, it is typically cheaper to compute $R = L^{-1}$ via the Cholesky factorization $LL^{\mathrm{T}} = I_m + Q_i^{\mathrm{T}}(2B+B^2)Q_i = Q^{\mathrm{T}}D^{\mathrm{T}}DQ$ whereas when $m \gg n_i$, it is typically more efficient to find the spectral decomposition $P\Lambda P^{\mathrm{T}} = Q_i^{\mathrm{T}}(2B + B^2)Q_i$ and then compute $R = I_p - P\bar{\Lambda}P^{\mathrm{T}}$ where $\bar{\Lambda} = (I_p + \Lambda)^{-1/2} + I_p$. In either case it follows that $\tilde{Q} := DQR$ is an orthogonal basis of $D(L)$.

Acknowledgements

Javier Peña's research has been funded by NSF grant CMMI-1534850.

References

[1] S. Agmon. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6(3):382–392, 1954.

[2] S. Ahipasaoglu, P. Sun, and M. Todd. Linear convergence of a modified Frank-Wolfe algorithm for computing minimum-volume enclosing ellipsoids. *Optimization Methods and Software*, 23(1):5–19, 2008.

[3] M. Baes. *Spectral Functions and Smoothing Techniques on Jordan Algebras: How algebraic techniques can help to design efficient optimization algorithms.* Lambert Academic Publishing, 2009.

[4] A. Basu, J. A De Loera, and M. Junod. On Chubanov's method for linear programming. *INFORMS Journal on Computing*, 26(2):336–350, 2013.

[5] A. Beck and S. Shtern. Linearly convergent away-step conditional gradient for non-strongly convex functions. *Math. Program., To Appear.*

[6] A. Belloni, R. Freund, and S. Vempala. An efficient rescaled perceptron algorithm for conic systems. *Math. of Oper. Res.*, 34(3):621–641, 2009.

[7] S. Chubanov. A strongly polynomial algorithm for linear systems having a binary solution. *Math. Program.*, 134:533–570, 2012.

[8] S. Chubanov. A polynomial projection algorithm for linear feasibility problems. *Math. Program.*, 153:687–713, 2015.

[9] G.B. Dantzig. An ϵ-precise feasible solution to a linear program with a convexity constraint in $\frac{1}{\epsilon^2}$ iterations independent of problem size. Technical report, Stanford University, 1992.

[10] J. Dunagan and S. Vempala. A simple polynomial-time rescaling algorithm for solving linear programs. *Math. Program.*, 114(1):101–114, 2006.

[11] M. Epelman and R. Freund. A new condition measure, preconditioners, and relations between different measures of conditioning for conic linear systems. *SIAM J. on Optim.*, 12:627–655, 2002.

[12] M. Epelman and R. M. Freund. Condition number complexity of an elementary algorithm for computing a reliable solution of a conic linear system. *Math. Program.*, 88(3):451–485, 2000.

[13] J. Faraut and A. Korányi. *Analysis on Symmetric Cones*. Oxford Mathematical Monographs, 1994.

[14] S. Lacoste-Julien and M. Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[15] D. Li, C. Roos, and T. Terlaky. A polynomial column-wise rescaling von Neumann algorithm. Technical report, Lehigh University, 2015.

[16] T. S. Motzkin and I. J. Schoenberg. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6(3):393–404, 1954.

[17] B. N. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall, Englewood Cliffs, NJ, 1980.

[18] J. Peña, D. Rodríguez, and N. Soheili. On the von Neumann and Frank-Wolfe algorithms with away steps. *SIAM J. on Optim.*, 26(1):499–512, 2016.

[19] J. Peña, V. Roshchina, and N. Soheili. Some preconditioners for systems of linear inequalities. *Optimization Letters*, pages 2145–2152, 2014.

[20] J. Peña and N. Soheili. A deterministic rescaled perceptron algorithm. *Math. Program.*, 155(1-2):497–510, 2016.

[21] C. Roos. An improved version of Chubanov's method for solving a homogeneous feasibility problem. Technical report, Delft University of Technology, 2015.

[22] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Cornell Aeronautical Laboratory, Psychological Review*, 65(6):386–408, 1958.

[23] S. Schmieta and F. Alizadeh. Associative and Jordan algebras, and polynomial time interior-point algorithms for symmetric cones. *Math. Oper. Res*, 26(3):543–564, 2001.

[24] S. Schmieta and F. Alizadeh. Extension of primal-dual interior point algorithms to symmetric cones. *Mathematical Programming*, 96(3):409–438, 2003.

[25] N. Soheili and J. Peña. A smooth perceptron algorithm. *SIAM J. on Optim.*, 22(2):728–737, 2012.

[26] N. Soheili and J. Peña. A primal–dual smooth perceptron–von Neumann algorithm. In *Discrete Geometry and Optimization*, pages 303–320. Springer, 2013.